ANO II - Nº 18 - Cr$ 155,00

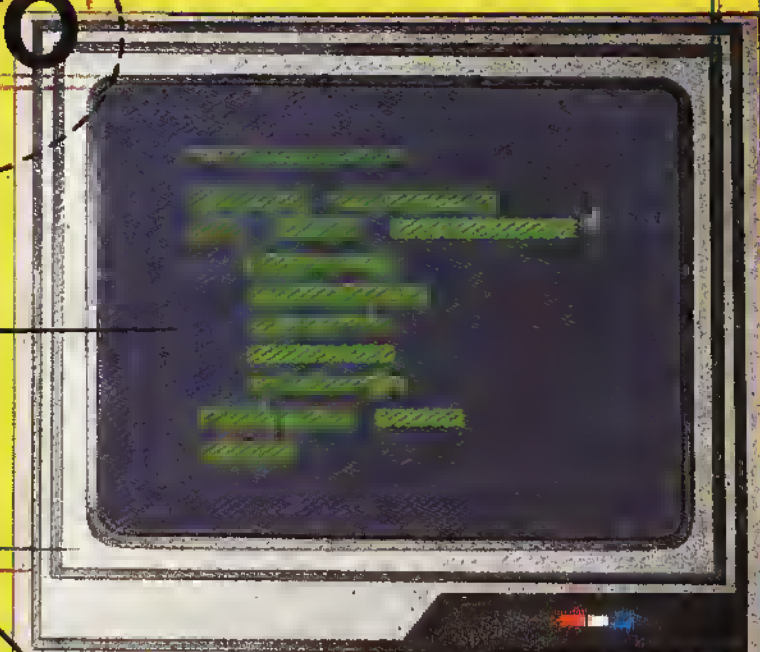
# PROJETO SPECTRO

# COBOL NO AMBIENTE MSX

# INTERFACE DE POTÊNCIA PARA MICROS

## LICENCE TO KILL
## F-16 FIGHTING FALCON

GO TO

?

Via Aérea - MANAUS, BOA VISTA, SANTARÉM, RIO BRANCO, JIPARANÁ, PORTO VELHO E AMAPÁ - Cr$ 201,50

**AGUIA INFORMÁTICA LTDA**
AV. N. S. DE COPACABANA 605/804
COPACABANA
22040 - RIO DE JANEIRO - RJ
TELEFONE: PABX(021) 235-3541
TELEX: 21.21717 KPUR BR

**DIRETOR RESPONSÁVEL**
GONÇALO R. F. MURTEIRA

**DIRETOR COMERCIAL**
JOSÉ IDEMAR A. NASCIMENTO

**DIRETORA DE JORNALISMO**
SOLANGE CALVO
MTB 19.524

**ADMINISTRAÇÃO**
LUZIMAR GOMES DA SILVA

**PUCLICIDADE**
VIVIAN PESSANO
MARY AZEREDO

**ASSINATURAS**
MONICA VICENTE

**COLABORADORES**
PAULO MARQUES FIGUEIRA
SERGIO GUY PINHEIRO ELIAS
PAULO ROBERTO PINHEIRO ELIAS
BRUNO MARRUT
JULIO VELLOSO
SERGIO DURIC CALHEIROS
GUILHERME A. L. DA SILVA
ANDRÉ L. A. SANTOS

**CAPA**
MILTON MEIER JUNIOR

**PRODUÇÃO**
WELLINGTON SILVARES
CARLOS AUGUSTO BAUER GUIMARÃES

**CIRCULAÇÃO**
ORLANDO NUNES

**COMPOSIÇÃO**
ALFA LÓGICA PROCESSAMENTO

**MONTAGEM**
GGM

**FOTOLITOS**
PROJETA STUDIO

**IMPRESSÃO**
EDITORA BRASILIANA

**DISTRIBUIÇÃO**
FERNANDO CHINACLIA DISTRIBUIDORA S.A.
Rua Teodoro da Silva, 907
Fone (021) 577-6655



# PAUTA

## ARTIGOS

COBOL NO AMBIENTE MSX ........................ 6
TESTE ANALÍTICO PARÂMÉTRICO
DE VARIÂNCIA EM DUAS PARTES .................. 8

## PROGRAMAS UTILITÁRIOS

NAS MALHAS DO COBOL ........................ 16
POR DENTRO DA INTERFACE DE DRIVE —
PARTE II ........................................ 48

## HARDWARE

INTERFACE DE POTÊNCIA PARA MICROS......... 14

## JOGO

MSX LORDS ........................................ 34

## SEÇÕES

NEWS ............................................. 4
ENTREVISTA ...................................... 42
CARTAS ........................................... 44
DICAS ............................................ 74

## PROJETOS

MSXDEBUG .38
SPECTRO .............................. .40

## SOS GAMES

F-16 FIGHTING FALCON ........................... 66
LICENCE TO KILL ................................. 68
JOE BLADE ........................................ 74

## ELEBRA COMEÇA PRODUZIR PEÇAS PARA SUA IMPRESSORA A LASER

Um investimento de US$ 600 mil é o que a Elebra Periféricos está movimentando em sua nova linha de montagem que será usada para iniciar a produção nacional de controladores e impressoras a laser. Os controladores , com capacidade de memória e processamento semelhantes a um micro de 32 bits, serão fabricados com tecnologia SMT (Surface Mounting Technology) — que eleva o grau de automação no processo de industrialização do produto.

O investimento da Elebra envolveu estudos de engenharia para adequar a produção dos controladores às condições brasileiras. Alguns itens do projeto foram redimensionados de acordo com os componentes disponíveis no mercado.

A Elebra continuará importando, para montagem das impressoras, apenas o "engine" (componente que comanda a impressão), da empresa japonesa Ricoh. O preço das impressoras deve permanecer inalterado no mercado.

## SACCO COLOCA WINCHESTER REMOVÍVEL NO MERCADO

A Sacco Computer vai comercializar no Brasil o winchester (disco rígido para armazenamento de dados) removível, produzido pela empresa norte-americana Plus Development Corporation em três modelos: com capacidade de 20 Mb, 40 Mb e 80 Mb (megabytes, milhões de caracteres).

O Plus Passport, como foi batizado, é um winchester composto de três partes: placa controladora, um suporte instalado no micro e um cartucho, onde ficam os discos.

Esse produto é indicado para cópias-reserva de informações, tornando a operação mais rápida. Basta instalar o Plus Passport que ele copia, em segundos, as informações do disco rígido do micro.

Além de centralizar informações de vários departamentos e poder ser aplicado em sistemas de redes locais, o winchester permite, também, que o usuário mantenha seu micro ligado a um sistema operacional multiusuário (como o Xenix) e, ao mesmo tempo, utilize aplicações guardadas no Plus Passport, que operam sob o sistema operacional DOS.

O Plus Passport tem um tempo médio de trabalho, sem defeitos, de 60 mil horas. Seu preço deve ficar na faixa dos US$ 5,5 mil e a Sacco dará assistência técnica para o equipamento.

A Sacco comercializará, além do winchester, o "hardcard", uma placa controladora de winchester que tem, em sua extremidade,um compartimento com o disco rígido

Uma das vantagens do "hardcard" é sua portabilidade. Caso o usuário viaje, é só abrir o micro, tirar a placa e colocá-la em uma pasta executiva. O "hardcard" é produzido em três modelos: 20Mb, 40 Mb e 80 Mb.

## ABC SOFTWARE

A ABC Software compra e vende aplicativos e utilitários para TK 90X, MSX 1.0 e 2.0 (350 e 720 Kb), e ainda troca jogos.

Informações pelo fone (011) 448-0953 ou à Rua Caçapava, 40, Baeta Neves, São Bernardo do Campo, São Paulo (CEP 09725).

## OMEGA CRIA MELHOR OPÇÃO EM SOFTHOUSE

A Omega Advanced Systems, empresa softhouse, trabalha em regime de "self-renovation", ou seja, todos os softs que tiverem mais de três anos de mercado ou que não tiverem boa aceitação junto aos usuários são apagados do seu arquivo. Esse controle faz com que o acervo da empresa seja um dos mais qualificados do país. Dentre todas as vantagens do sistema, a Omega oferece, ainda, garantia de dois anos sobre o disquete e seu conteúdo.

Informações pelo telefone (011) 522-2613.

# COBOL NO AMBIENTE MSX

MÁRCIO MACHADO DE MOURA

O uso da linguagem COBOL em equipamentos MSX é um fato muito pouco divulgado e visto de maneira até um pouco cética por parte de alguns programadores. Porém, já é uma realidade que, embora como dito, sem divulgação, apresenta performances excelentes, que possibilitam o uso de sistemas comerciais no padrão MSX, que antes eram exclusivos de equipamentos mais conhecidos para este fim, e também mais caros.

Tenho desenvolvido, no decorrer dos últimos anos, sistemas na linguagem COBOL para linha MSX, destinados aos mais diversos campos da aplicação comercial como: Estoque, Contabilidade, Fluxo de Caixa, etc. Garanto que a performance de tais sistemas mudou, em muito, a opinião dos usuários que tiveram contato com eles, principalmente sobre o que um equipamento MSX pode fazer.

## PRODUÇÃO DE PROGRAMAS

Para produção de programas e sistemas em liguagem COBOL, utilizando-se para tal de um equipamento padrão MSX, é necessário um conjunto de arquivos que formam o pacote da Microsoft, designado COBOL 80. Os arquivos necessários para essa produção, são os seguintes:

| COMPILAÇÃO | LINKEDIÇÃO |
|---|---|
| COBOL .COM | L80 .COM |
| COBOL1 .OVR | COBLIB .REL |
| COBOL2 .OVR | 1CRTDRV .REL |
| COBOL3 .OVR | |
| COBOL4 .OVR | |

Existem, ainda, outros arquivos dedicados ao suporte da produção de softwares, mas que não são especificamente associados ao COBOL, embora possam ser utilizados em conjunto. Chegaremos lá em outra oportunidade, mas antes, temos muito o que falar apenas sobre o COBOL.

A produção de um programa executável, com base em um fonte escrito na linguagem COBOL, passa por duas fases: Compilação e Link-edição. Na primeira fase, obtemos um arquivo objeto, resultado da transformação do texto fonte em linguagem de máquina. Nesta fase também são detectados todos os erros de sintaxe, que são listados com a indicação da linha, o elemento não identificado e o tipo de erro cometido. Em seguida temos a fase de Link-edição que liga, ao arquivo objeto, todas as rotinas da biblioteca COBOL, assim como as rotinas básicas de tratamento de teclado e video.

Deve-se tomar muito cuidado com um arquivo L80.COM, que anda rodando pelo meio dos usuários do padrão MSX, pois o mesmo não funciona. O original, ou pelo menos um que tenho certeza que funciona, deve ter o tamanho de 11.264 bytes. Além da biblioteca básica das rotinas em COBOL (COBLIB.REL), temos a biblioteca de tratamento de teclado e video (CRTDRV.REL), que já existe em versões dedicadas especificamente ao padrão MSX.

O sistema COBOL 80 não tem editor próprio, sendo necessário portanto, o uso de um editor de textos para a produção dos programas. O formato do arquivo, gerado pelo editor, deve ser padrão de texto do sistema (formato ASCII), tomando cuidado com alguns editores que colocam bytes de controle nas linhas de edição. Um editor, fácil de ser encontrado no mercado, e com um tamanho razoável para ser colocado no mesmo disco do COBOL, é o SCED, do pacote DUAD da ASCII Corporation.

A produção de arquivos executáveis, passa pelo seguinte algoritimo:

1 — Edição do programa fonte;
2 — Compilação [geração do arquivo objeto];
3 — Depuração dos erros (Retorna a [1] se houve erro);
4 — Link-edição [geração do arquivo executável].

Os arquivos envolvidos neste algoritimo são os seguintes:

• PROG.COB — Programa em COBOL [Fonte]
• PROG.REL — Resultado da Compilação [Objeto]
• PROG.COM — Resultado da Link-edição [Executável]

Podem existir arquivos com extensão LST ou PRN, resultado da compilação, que são dedicados a documentação (linhas numeradas, status de erro, etc.), porém não são obrigatórios.

O tamanho máximo de um executável, gerado pelo pacote COBOL 80, é de aproximadamente 42 Kbytes, o que corresponde, dependendo das rotinas utilizadas, a mais de mil linhas de programa.

## COMPILAÇÃO DOS PROGRAMAS

A execução do compilador COBOL pode ser feita de modo direto ou indireto, ou seja, podemos compilar um programa usando atributos ao lado do comando COBOL, ou executando o programa COBOL, e em seguida passando os parâmetros de compilação. A forma geral do uso direto é a seguinte:

A>COBOL NOME2,SAÍDA=NOME1
Onde:
NOME1 = Nome do programa fonte com extensão COB;
NOME2 = Nome que se deseja para o arquivo objeto;
SAÍDA = Listagem documentada do programa.

Na SAÍDA pode ser colocado tanto um nome de arquivo, para efeito de documentação, como um periférico de saída:

TTY: = Video;
LST: = Impressora;

O símbolo ortográfico **dois pontos,** faz parte do nome do periférico. Outro detalhe quanto à saída, é o fato dela ser opcional, isto é, pode ser excluida do comando. Como exemplos das formas possiveis de compilação, podemos citar:

1 — COBOL PROG,TTY:=PROG
2 — COBOL PROG,LST:=PROG
3 — COBOL PROG,DOC1=PROG
4 — COBOL TESTE=CADASTRO

No primeiro exemplo, temos a compilação do programa PROG.COB, com a geração do arquivo PROG.REL, mostrando, no video, o programa conforme é feita a compilação.

O segundo exemplo é idêntico ao primeiro, com a diferença de gerar a listagem na impressora em vez do video.

No terceiro exemplo, além do objeto PROG.REL, um outro arquivo é gerado, com o nome de DOC1, contendo a listagem numerada do programa, assim como o status de erro.

Finalmente no quarto exemplo, temos a compilação do programa CADASTRO.COB, com a geração do objeto TESTE.REL, sem que nenhuma saída documentada seja gerada.

O uso do compilador COBOL, executando apenas o arquivo COBOL.COM, é idêntico a forma já apresentada, exceto no fato de ser apresentado um asterisco, que

funciona como prompt do COBOL, por onde devem ser passados os parâmetros de compilação. Exemplo:

```
A>COBOL
*PROG,TTY:=PROG
```

## LINK-EDIÇÃO DOS PROGRAMAS

O uso do link-editor L80, é semelhante ao do compilador, podendo ser usado de maneira direta, ou com a execução do arquivo L80.COM, com a passagem posterior dos parâmetros de linkedição. A forma geral da utilização do linkeditor é a seguinte:

```
A>L80 NOME2/N,NOME1/E
```

O rótulo NOME1 identifica o nome do arquivo objeto, obtido pelo processo de compilação e deve ter a extensão REL. O rótulo NOME2 representa o nome do arquivo executável, gerado pelo linkeditor. Os comandos, após as barras, representam as seguintes funções:

**/N** — Identifica o nome do arquivo executável;
**/E** — Indica que deve ser gerado o executável.

Um comando **/G**, pode ser usado em lugar da expressão **/E**, que além de indicar que o executável deve ser gerado, executa o programa automaticamente.

No uso indireto do linkeditor, existem ainda outros comandos, como por exemplo a expressão **/U**, que lista as rotinas pendentes no programa, que não foram identificadas nas bibliotecas do COBOL. Conforme formos avançando na nossa explanação sobre COBOL, outros macetes do uso do linkeditor serão apresentados.

## ESTRUTURA DA LINGUAGEM

Os programas em COBOL se dividem em quatro áreas que controlam todas as funções internas do programa. As áreas, com suas respectivas funções, são as seguintes:

**IDENTIFICATION DIVISION**: Área dedicada à identificação do programa, onde são colocadas informações gerais como: nome, data de criação, equipamento utilizado, etc.

**ENVIRONMENT DIVISION**: É a divisão de ambiente, onde são descritos os nomes lógicos e tipos dos arquivos utilizados, assim como o formato dos números decimais (com vírgula decimal ou ponto decimal).

**DATA DIVISION**: Na divisão de dados, todas as variáveis utilizadas pelo programa são identificadas da mesma maneira que é descrita a estrutura de dados de qualquer arquivo definido na ENVIRONMENT DIVISION, inclusive os nomes físicos (assumidos pelo sistema operacional) dos arquivos. Também temos, nessa área, uma seção de tratamento de vídeo, denominada de SCREEN SECTION, onde podem ser formatadas telas utilizadas pelo programa.

**PROCEDURE DIVISION**: Nesta área, temos o programa propriamente dito, onde são descritas as rotinas que compõem todas as linhas de instrução do programa.

Notem que a linguagem COBOL é muito organizada, gerando programas de fácil leitura e documentação, principalmente pelo fato de termos linhas de instrução quase que em inglês corrente, facilitando assim a depuração da lógica do programa.

O tratamento de cada área é feita de maneira independente, no momento da compilação, existindo um overlay para cada divisão. Esse controle é feito de maneira transparente ao usuário, havendo também a geração de um arquivo de controle que é apagado do diretório, após os procedimentos de compilação.

O tempo necessário para a produção de um programa executável, a partir de um fonte em COBOL, varia de acordo com o tamanho do programa, podendo chegar até a 15 minutos, em um drive de 3"½, com programas de mais de mil linhas.

## CONCLUSÃO

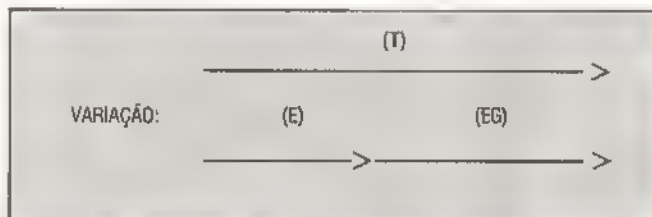
Analisados os principais aspectos da produção de programas em COBOL, passaremos nos próximos artigos para a descrição da linguagem propriamente dita, apresentando rotinas e técnicas de programação, até alcançarmos uma compreensão suficiente do uso dessa linguagem, para que possamos apresentar sofisticadas técnicas que envolvem a mistura de programas em COBOL, com rotinas em C e ASSEMBLER.

# TESTE ESTATÍSTICO PARAMÉTRICO DA ANÁLISE DE VARIÂNCIA EM DUAS PARTES

## INTRODUÇÃO

Um dos testes estatísticos mais conhecidos e de grande aplicação na avaliação comparativa de 2 grupos de dados (ou de variáveis) provenientes de pesquisas biológicas é, sem dúvida, o teste "t" de Student. Neste teste, uma população de dados, que estão incluídos na chamada Curva de Distribuição de Gauss (Curva Normal), serve de parâmetro para outra população de dados, daí o termo "teste paramétrico". Dentre estes tipos de teste, sobressai-se a Análise de Variância, conhecida também pela sua sigla em inglês, ANOVA (ANalysis Of VAriance), ou, ainda, como Análise de Variância de Fisher (1).

A análise de variância é o método estatístico ideal para comparar 3 ou mais agrupamentos de dados. Nela, o parâmetro empregado para uma estimação global destes agrupamentos é o que se denomina VARIAÇÃO, ou seja, a SOMA DOS QUADRADOS dos AFASTAMENTOS (ou DESVIOS) da média dos dados digitados. Ao serem calculadas a nível de quadrados, as diferenças entre os grupos analisados são muito mais evidentes. Por isso, a análise de variância de Fisher é muito mais sensível que o teste "t" de Student, o qual é baseado no simples valor do erro-padrão ("Standard Deviation").

Representando o valor dos quadrados dos desvios por vetores, temos as variações: total (T), entre grupos (EG) e erro (E), como se segue:

Por este gráfico, vemos que a variação total foi DECOMPOSTA em DUAS partes, uma das quais (EG) é de interesse para a análise pretendida. As variâncias são o quociente resultante da divisão das variações pelos respectivos graus de liberdade. Estes, em última análise, correspondem ao número de desvios da média, número este escolhido arbitrariamente, com exceção de um, o qual deverá ser determinado para tornar nula a soma desses desvios. É, portanto, o número de dados digitados para cada tipo de variação (T, EG e E), menos 1 unidade. Por exemplo, para um total (T) de 20 dados digitados, o grau de liberdade será de 20–1=19. Isto se trata de um artifício para tornar "Gaussiana" a Curva dos desvios da média determinados.

A inferência estatística é dada pelo valor de "F", um valor igual ao quociente resultante da divisão entre a variância entre grupos (EG) e aquela relativa ao erro (E). O valor de F determinado é comparado com os dados de uma Tábua de F para uma significância de 5% e 1%. Se o nosso F experimental for maior do que um dos 2 valores de F da Tábua, a variação em estudo terá significância para $p < 0.05$ ou $p < 0.01$, respectivamente.

## PROCEDIMENTO DA ANÁLISE PELO PROGRAMA DESTE ARTIGO

O usuário deverá informar, inicialmente, o número de grupos (agrupamentos de dados ou "tratamentos") a serem analisados. A seguir, é oferecida a oportunidade de que esta digitação possa ser feita com recursos de correção de dados dentro de cada grupo. O programa informa, antes de cada entrada de dados, o grupo e a ordem a que os mesmos se referem. Caso um determinado valor tenha sido digitado incorretamente, na entrada de dados seguinte poder-se-á digitar "R" ou "r" e assim retroceder ao dado anterior. Este é exibido na mesma linha de digitação, com o objetivo de informar ao usuário se o erro foi realmente cometido. Em caso negativo, bastará teclar < return > para manter o valor anteriormente digitado.

Ao terminar de digitar valores de um grupo, teclando-se "T" ou "t", passa-se ao grupo seguinte, ou, se o grupo anterior tiver sido o último, ao relatório do processamento de cada grupo, incluindo os dados digitados, que assim poderão ser devidamente conferidos.

A cada relatório de grupo, o usuário tem a opção de passar à impressora as mesmas informações da tela, podendo ou não incluir os dados digitados. Estas informações constam de:

a – total de dados do grupo;
b – soma dos dados do grupo e a sua média;
c – média dos dados;
d – quadrado da soma;
e – quadrado da soma/nº de dados do grupo;
f – soma dos quadrados.

Se forem muitos os dados digitados, é bem possível que o relatório parcial não caiba totalmente na tela, provocando o rolamento ("scrolling") da impressão. Quando isto acontecer, teclando-se < stop > pode-se parar momentaneamente esta rolagem, para que os dados possam ser lidos.

Terminados os relatórios parciais acima citados, obtém-se o relatório final, o qual corresponde à análise de variância propriamente dita: a variação (total, entre grupos e erro), as respectivas somas dos quadrados e graus de liberdade, e final-

# TRABALHOS DIFERENTES. FERRAMENTAS DIFERENTES. TUDO DISCOVERY.

A DISCOVERY INFORMATICA possui a mais completa linha de softwares para o seu MSX. Qualquer que seja a sua necessidade, nós possuimos a ferramenta ideal para o seu trabalho, com preço justo e desempenho acima do mercado. Porém, por mais diferentes que sejam, os nossos softwares possuem uma coisa em comum: a qualidade DISCOVERY, que já se tornou sinônimo de competência. Adquira e comprove. Seu conceito sobre software nacional nunca mais será o mesmo.

## DESKTOP PUBLISHING

A Discovery preparou para você três programas que incrementarão ainda mais seu MSX. Todos com a mesma facilidade de operação do já conhecido PROFESSIONAL PUBLISHER. Programas especificos para aplicações específicas.

### PROFESSIONAL LABELS

Cria etiquetas personalizadas em geral. Excelente para você decorar seus disquetes, fitas de video, livros, cadernos e tudo mais que sua imaginação desejar. Permite a impressão em uma ou duas colunas, com quantas cópias você quiser. Permite armazenar a etiqueta para uso posterior. Total compatibilidade com os produtos da coleção Desktop Publishing da DISCOVERY. Compatível com shapes e alfabetos do editor Graphos 3. Se você gosta das etiquetas personalizadas das softhouses, agora você também poderá criar a sua! Disponivel em 5 1/4 ou 3 1/2 pols. **PREÇO: 30 BTNFs.**

### PROFESSIONAL STRIPES

Você já foi numa festa de aniversário e encontrou uma daquelas faixas enormes, feita no computador ? Ou já viu alguma faixa anunciando uma liquidação ? Pois é, agora você poderá usar seu MSX para criar faixas com até 4,60 m de comprimento ! Impressão com quatro padronagens diferentes para economia da fita. Armazenamento das faixas para uso posterior, compatibilidade com Graphos 3. Precisa falar mais alguma coisa ? Disponivel em 5 1/4 ou 3 1/2 pols. **PREÇO: 30 BTNFs.**

### PROFESSIONAL CARDS

Não gaste mais dinheiro comprando cartões comemorativos. Aproveite seu dinheiro para comprar mais presentes no Natal, ou no aniversário da pessoa querida. Este programa permite criar cartões comemorativos nos formatos mais comuns. Compatível com Graphos 3. Permite armazenar os cartões para uso posterior. Seja uma autêntica papelaria! Crie tantos cartões quanto sua imaginação permitir. Disponivel em 5 1/4 ou 3 1/2 pols. Acompanha um disco com shapes comemorativos especialmente selecionados da coleção Desktop Publishing. **PREÇO: 40 BTNFs.**

## E MAIS:

### PROFESSIONAL PUBLISHER

O melhor editor para Desktop Publishing do mercado. Permite criar páginas como as elaboradas nos melhores programas para a linha PC. Permite o uso da Megaram-Disk e é capaz de criar desde jornais até pequenos informativos. Disponivel em 5 1/4 ou 3 1/2 pols. AUTOR: Vitor Hugo P. Costa. **PREÇO: 120 BTNFs.**

### PROFESSIONAL PUBLISHER ADVANCED

O melhor editor agora é o mais completo. Todos os programas acima descritos reunidos num só produto. Crie páginas, etiquetas personalizadas, faixas e cartões sem trocar de programa. Uma verdadeira Work-Station! **PREÇO: 210 BTNFs.**

**ATENÇÃO:** Todos os programas são independentes quando comprados em separado. Você também poderá adquirí-los aos poucos, para formar depois o sistema ADVANCED.

### USUÁRIOS DO PROFESSIONAL PUBLISHER

Para usuários cadastrados do PROFESSIONAL PUBLISHER que desejem adquirir cada módulo separadamente, terão um desconto de 30 %. Se o usuário cadastrado desejar o sistema ADVANCED poderá requerer atualização por 65 BTNFs.

A atualização só poderá ser feita diretamente à DISCOVERY INFORMATICA. Envie o número da cópia do seu PROFESSIONAL PUBLISHER junto com seu pedido, seja de algum módulo ou do sistema ADVANCED.

Para fazer o pedido, envie cheque nominal ou vale postal (ag. Primeiro de Março) à:

DISCOVERY INFORMATICA LTDA
RUA DA QUITANDA 19 SL 404
CENTRO - RIO DE JANEIRO - RJ
CAIXA POSTAL 3043 - CEP 20001

mente a variância. Abaixo desta tabela, é mostrado o valor de F, que servirá de guia para a avaliação da significância.

Os valores da tábua de F (2) não constam do programa apresentado na Figura 2. A procura do valor de referência na Tábua se faz da seguinte forma: procura-se localizar na abcissa o grau de liberdade correspondente á variância maior observada entre as variações "entre grupos" e "erro", e, na ordenada, o grau de liberdade da variância menor entre as variações citadas: fazendo-se coincidir o prolongamento dessas coordenadas, achamos os valores da Tábua de F para 5% e 1%, este último geralmente impresso em itálico.

## O PROGRAMA

A análise de variância obriga ao usuário sem os recursos do computador a manipulação cansativa e, portanto, pouco confiável, de valores de grande número de dígitos, geralmente com o auxílio de uma calculadora, sem poder, na maioria das vezes, conferir os dados introduzidos. Assim, adquire-se óbvia vantagem em deixar esta tarefa por conta de um programa construído para esta finalidade, como aquele que mostramos na Figura 2.

O programa utiliza um algorítmo de entrada de dados através de uma variável alfanumérica, para depois transportar o dado digitado á sua posição correspondente no grupamento desejado. O objetivo deste método é o de facilitar as operações de retorno a valores anteriores e trocas de grupos, através de letras apropriadas. Uma matriz dimensionada em 10 x 180 elementos é definida no início do programa, para possibilitar a introdução de 10 grupos com um total de 180 dados em cada grupo. Foi necessária a diminuição de 200 para 180 elementos nesta matriz, pelo fato particular de estarmos operando com um filtro de impressão alojado no endereço &HD000. Se no seu caso, leitor, este filtro não for necessário, você poderá redefinir a matriz para o total de 200 elementos. Como as condições de memória do MSX são bastante críticas na área do Basic de Disco, poderá ser necessário fazer alguns ajustes com relação ao tamanho desta matriz (comando DIM) ou à reserva de Bytes na memória para armazenar todos os dados (comandos CLEAR).

O digitador deverá prestar atenção na digitação dos dados, para não entrar com letras ou outros caracteres, o que invalidará esta entrada. Caso haja alguma dúvida a este respeito, retroceda ao valor digitado suspeito e será exibido o valor anteriormente digitado ou aguarde a emissão do respectivo relatório.

Uma crítica á entrada de dados é feita através do teste do contador que identifica a posição de cada dado na matriz. Quando o valor acumulado por este ultrapassar o máximo admitido em cada grupo, o programa passa para o grupo seguinte ou então encerra a digitação, passando aos relatórios iniciais. Caso o tamanho da matriz seja alterado, lembre-se de alterar também a linha onde este teste é realizado.

O cálculo relativo aos relatórios de cada grupo, onde estão incluídos os valores da soma dos dados e de seus respectivos quadrados, está disposto em dois "loops" (laços) sobrepostos em técnica de ninho (um loop dentro do outro). Desta maneira, é possível coletar e processar os dados que interessam de um grupo e juntá-los aos dados obtidos dos outros grupos, os quais são mais tarde aproveitados no cálculo final das variâncias. O esquema deste algoritmo está mostrado na Figura 1.

Para encurtar a redação do programa, optamos por utilizar o mesmo algorítmo de impressão de dados, tanto na tela quanto na impressão. No BASIC do MSX, isto é possível abrindo-se um arquivo de saída, tratando estes periféricos como dispositivos (CRT: e LPT:; respectivamente). Neste caso, alguns comandos do BASIC relativos á formatação de tela (CRT:), tais como: LOCATE (X, Y) ou TAB(N) se tornam inadequados: o primeiro por não operar na impressora e o segundo por não levar em consideração o número de caracteres impressos previamente, oriundas de uma variável, quando se imprime mais de uma variável na mesma linha, desalinhando a montagem de uma tabela, como aquela que aparece em nosso relatório final. Este problema pode ser contornado de forma simples, utilizando-se funções de manipulação de variáveis. No nosso caso, as seguintes funções foram usadas: STRING$ (código ASCII do caracter número de posições impressas), SPACE$ (N), LEN (variável alfanumérica)e STR$ (variável numérica). Quando mais de uma variável é impressa na mesma linha, as seguintes fórmulas podem ser empregadas para a segunda variável em diante:
SPACE$ (X – LEN (variável) – para as variáveis alfanuméricas, ou SPACE$ (X – LEN (STR$ (variável))) – para as variáveis numéricas, onde X é a posição a partir da qual se deseja imprimir a variável e a variável especificada pela função LEN e LEN (STR$) é aquela posicionada imediatamente à esquerda.

O programa faz uso pleno da digitação em tela de 80 colunas. Da mesma forma, todas as mensagens e relatórios são exibidos nesta formatação. Havendo necessidade de usar o programa na screen de 40 colunas, as devidas adaptações deverão ser efetuadas, com o devido cuidado para não destruir o formato da tabela final impressa.

Todos os valores resultantes dos cálculos aparecerão nos relatórios sem arredondamentos, ficando para o usuário a decisão de efetuá-los ou não.

## UM EXEMPLO PARA CONFERIR A DIGITAÇÃO DO PROGRAMA

O teste servirá para se conferir se o programa, depois de digitado, está rodando corretamente.

Os dados abaixo representam os pontos obtidos por 48 ciclistas, durante uma corrida para determinar se havia diferença de performance entre 3 marcas de bicicletas. Os 48 ciclistas foram divididos em 3 grupos iguais (I, II e III), por escolha casual, cabendo a cada grupo uma das marcas de bicicleta.

Se não houver desvantagem de performance entre uma marca de bicicleta sobre qualquer uma das outras e se os pontos da média das populações correspondentes às 3 marcas forem simbolizadas por m1, m2 e m3, o problema se reduz a testar a hipótese (ho):

$$ho = m1 = m2 = m3$$

| I | II | III | I | II | III |
|---|---|---|---|---|---|
| 44 | 40 | 54 | 43 | 38 | 45 |
| 45 | 41 | 53 | 42 | 39 | 46 |
| 39 | 37 | 50 | 56 | 51 | 56 |
| 38 | 36 | 41 | 55 | 50 | 55 |
| 33 | 38 | 40 | 47 | 45 | 49 |
| 34 | 39 | 39 | 48 | 46 | 48 |
| 56 | 53 | 55 | 58 | 50 | 54 |
| 57 | 52 | 54 | 59 | 59 | 56 |

**RESULTADOS**

DADOS DIGITADOS NO GRUPO I
44 45 39 38 33 34 56 57 43 42 56 55 47 48 58 59

RELATÓRIO DO GRUPO 1

Total dos dados: 16
Soma dos dados: 754
Média dos dados: 47.125
Quadrado da soma: 568516
Quadrado da soma/nº de dados: 35532.25
Soma dos quadrados: 36688

DADOS DIGITADOS NO GRUPO 2
40 41 37 36 38 39 53 52 38 39 51 50 45 46 50 59

RELATÓRIO DO GRUPO 2

Total de dados: 16
Soma dos dados: 714
Média dos dados: 44.625
Quadrado da soma: 509796
Quadrado da soma/nº de dados: 31862.25
Soma dos quadrados: 32612

DADOS DIGITADOS NO GRUPO 3
54 53 50 41 40 39 55 54 45 46 56 55 49 48 54 56

RELATÓRIO DO GRUPO 3

Total de dados: 16
Soma dos dados: 795
Média dos dados: 49.6875
Quadrado da soma: 632025
Quadrado da soma/nº de dados: 39501-5625
Soma dos quadrados: 40027

**RELATÓRIO FINAL:**
**ANÁLISE DE VARIÂNCIA (ONE-WAY)**

| VARIAÇÃO | SOMA DOS QUADRADOS | GRAUS DE LIBERDADE | VARIÂNCIA |
|---|---|---|---|
| Total | 2635.97916667 | 47 | 56.084663120638 |
| Entre grupos | 205.04166667 | 2 | 102.520833335 |
| Erro | 2430.9375 | 45 | 54.020833333333 |

Valor de F: 1.8978017740378

No nosso exemplo, o grau de liberdade da variância menor é de 45 e o grau de liberdade da variância maior é 2. Com essas coordenadas (45 e 2), achamos, na Tábua de F, valores para 5% e 1% de probabilidade correspondentes a 3.21 e 5.12, respectivamente. Constatamos, assim, que o valor experimental de F não tem significância, visto que é menor do que o da Tábua para 5% de probabilidade.

Conclusões: baseados nesta análise, deduzimos que não há diferença de performance entre as 3 marcas de bicicletas.

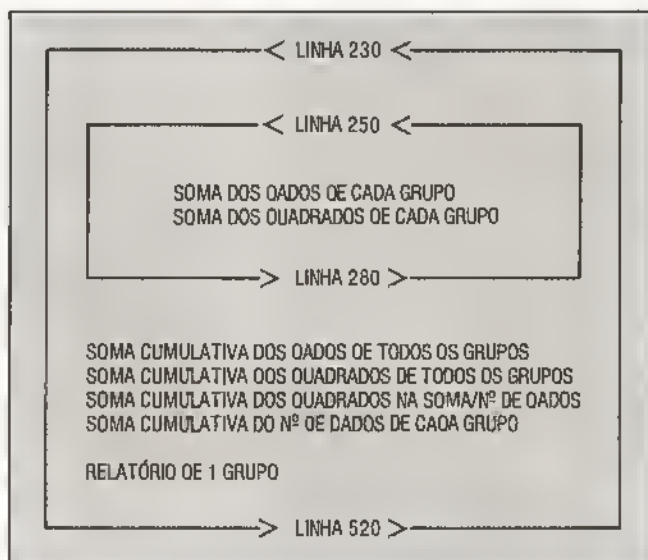



FIGURA 1 - Algoritmo para o cálculo da análise de variância em 2 loops dispostos em técnica de ninho

## SOBRE OS AUTORES


LOUIS BARRUCAND é Professor Titular e Livre Docente, enquanto que PAULO ROBERTO PINHEIRO ELIAS é Professor Adjunto e Mestre de Ciências. Ambos desenvolvem atividades didáticas e de pesquisa no Departamento de Patologia da Faculdade de Medicina da Universidade Federal do Rio de Janeiro.


**REFERÊNCIAS BIBLIOGRÁFICAS**


1 – Fischer, R.A. Statistical methods for research workers. Oliver & Boyd, ed. Edingburgh, 5th edition (1934).

2 – Snedecor, G.W. Statistical methods. Iowa State College Press, Ames, Iowa, 4th edition (1946).

```
I0 REM ANALISE DE VARIANCIA TIPO ONE WAY
20 REM PARA DIGITAÇÃO EM TELA DE 80 COLU
NAS
30 REM AUTOR: PAULO ROBERTO P. ELIAS
40 REM DATA: OUTUBRO/89
50 CLS:CLEAR 500
60 DIM A(10,I80),NC(I0)
70 C=0:SD=0 TQ=0.SV=0·Y=0
80 PRINT"ANALISE DE VARIANCIA DE DUAS PA
RTES (ONE-WAY):":PRINT
90 INPUT"Quantos grupos deseja analizar
";G
I00 IF G<2 OR G>10 THEN PRINT "Número de
 grupos inválido":PRINT.GOTO 90
II0 PRINT PRINT:PRINT"ENTRADA DE DADOS:"
:PRINT
I20 FOR I=1 TO G
130 IF C>I80 THEN C=C+I ELSE"Número máxi
mo de amostras ultrapassado· entrada enc
errada":G=G-1:NC(I)=C:C=0:GOTO 2I0
I40 PRINT:PRINT"DIGITE: <CR> (valor anti
go), R(etroceder), T(muda de grupo/encer
ra digitação)"
150 PPINT"GRUPO N' ";I;" ITEM ";C; " Val
or antigo: ";A(I,G);" ";.INPUT"Novo valo
r: ";NV$
I60 IF NV$="R" OR NV$="r" THEN IF C>2 TH
EN C=I:NV$="":GOTO I40 ELSR C=C-1:NV$=""
:GOTO I40
I70 IF NV$="T" OR NV$="t" THEN C=C-1:NC(
I)=G:C=0:PRINT·NV$="":GOTO 2I0
180 IR NV$="" THEN 200
I90 A(I,C)=VAL(NV$)
300 GOTO 130
2I0 NEXT I
220 PRINT·PRINT"AGUARDE     "
230 FOR I=1 TO G
240 X=0·SQ=0
250 FOR F=I TO NC(I)
260 X=X+A(I,F)
270 Q=A(I,R)^2:SQ=SQ+Q
280 NEXT F
290 SD=SO+X:TQ=TQ+SQ
300 VA=X^2/NC(I):SV=SV+VA
3I0 Y=Y+NC(I):ME=X/NC(I)
320 CLS:OPEN "CRT:" FOR OUTPUT AS #I
330 PRINT#I,"DADOS DIGITADOS NO GRUPO";I
:PRINT#1,
340 FOP P=I TO NC(I)
350 PRINT#I,A(I,P);" ";
360 NEXT P:PRINT#I,:PRINT#I,
370 PRINT#I,"RELATORIO DO GRUPO";I·PRINT
#I,
380 PRINT #1,"TOTAL DE DADOS: ":NC(I)
390 PRINT#I,"SOMA DOS DADOS: ";X
400 PRINT#1,"MéDIA DOS DADOS";ME
4I0 PRINT#I,"QUADRADO DA SOMA  ";X^2
420 PRINT#I,"QUADRADO DA SOMA/Nº DO GRUP
O: ";VA
430 PRINT#I,"SOMA DOS QUADRADOS: ";SQ
440 PRINT#I,:PRINT#I,:PRINT#I.
450 CLOSE#I
460 IF I$="S" OR I$="s" THEN I$=""·D$=""
:GOTO 500 ELSE PRINT:PRINT"SAIDA PARA IM
PRESSORA (S/N) ?";I$=INPUT$(1)
470 IF I$="S" OR I$="s" THEN OPEN "LPT:"
 FOR OUTPUT AS #I·GOTO 480 ELSE IF I$="N
" OR I$="n" THEN 5I0 ELSE 450
480 PRINT:PRINT"INCLUI DADOS DIGITADOS (
S/N) ?";D$=INPUT$(I)
490 PRINT·PRINT"PREPARE A IMPRESSORA E T
ECLE ALGO";T$=INPUT$(1);IF D$="N" OR D$=
"n" THEN 370 ELSE 330
500 IF I<G THEN PRINT:PRINT"TECLE ALGO P
ARA O RELATORIO DO PROXIMO GRUPO"ELSE PR
INT:PRINT"TECLE ALGO PARA O RELATORIO RI
NAL"
5I0 T$=INPUT$(I)
520 NEXT I
530 PRINT:PRINT"AGUARDE ..."
540 EI=SD'2/Y:E2=SV-E1:RR=TQ-SV:TI=TQ-RI
550 GI=Y-I;G2=G-I:G3=G1-G2
560 VI=T1/G1:V2=E2/G2:V3=ER/G3
570 R=V2/V3
580 CLS:OPEN "CRT:" FOR OUTPIIT AS #I
590 PRINT #I,"RELATÓRIO RINAL· ANALISE D
E VARIANCIA (ONE-WAY)":PRINT#1,:PRINT#I.
600 PRINT#I,"VARIAÇAO";SPACE$(6);"SOMA D
OS QUADRADOS",SPACE$(3);"GRAUS DE LIBERD
ADE";SPACE$(7);"VARIANCIA"
610 PRINT#I,STRING$(79,45)
620 PRINT#I,"TOTAL";SPACE$(8);TI;SPACE$(
27-LEN(STR$(TI)));GI;SPACE$(I7-LEN(STR$(
GI)));VI
630 PRINT#I."ENTRE GRUPOS";SPACE$(I),E2;
SPACE$(27-LEN(STR$(E3)));G2,SPACE$(I7·LE
N(STR$(G2))),V2
640 PRINT#I,"ERRO",SPACE$(9);ER;SPACE$(2
7-LEN(STR$(ER)));G3;SPACE$(I7-LEN(STR$(G
3)));V3
650 PRINT#I.STRING$(79,45)
660 PRIT#I,SPACE$(35);"Valor de F· ";F
670 PRINT#I,:PRINT#1,:PRINT#1,:PRINT#I,
680 CLOSR#I
690 IF I$="S" OR I$="s" THEN I$=""·GOTO
740
700 LOCATE,I5:PRINT"SAIDA PARA IMPRESSOR
A (S/N) ?" I$=INPUT$(I)
710 IF I$="S" OR I$="s" THEN 720 ELSR IF
 I$="N" OR I$="n" THEN 740 ELSE 700
720 OPEN "LPT:" FOR OUTPUT AS #I
730 LOCATE,17:PPINT"PREPARE A IMPRESSORA
 E TECLE ALGO":T$=INPUT$(I).GOTO 590
740 LOCATE,19:PRINT"DESEJA OUTRA ANALISE
 (S/N) ?      ":V$=INPUT$(I)
750 IF V$="S" OR V$="s" THEN CLS:GOTO 70
760 END
```

# INTERFACE DE POTÊNCIA PARA MICROS

Tanios Hamzo

Uma das aplicações mais importantes — e talvez uma das menos exploradas dos microcomputadores — é a sua capacidade de controlar processos e sistemas externos. A atividade normal que a maioria das pessoas logo imagina para os micros é aquela em que ficam estáticos sobre as mesas dos escritórios calculando, imprimindo e às vezes até se comunicando. O que pouca gente sabe é que essas poderosas ferramentas de trabalho também são capazes de fazer serviços mais pesados do que puxar formulários contínuos.

Sistemas de segurança, afazeres domésticos, temporizadores, monitoradores e uma infinidade de outras tarefas "braçais" são algumas aplicações que um micro qualquer pode efetuar, desde que tenha um estágio ou módulo de potência que sirva de ligação (interface) entre a seção lógica interna do micro e o aparelho ou dispositivo de potência a ser controlado.

Temos nesta matéria uma das melhores e mais simples formas de potencializar um microcomputador, dando-lhe capacidade para aplicações de controle de potência.

Partindo de uma saída lógica tipo TTL (que pode ser obtida de um dos pinos da comunicação paralela, por exemplo), alimenta-se o RELÊ K1 (um pequeno RELÊ ativável por uma saída TTL de 5 V). Os contatos deste relê fornecem uma corrente de ativação da porta do triac O1. Ao triac, por sua vez, cabe a tarefa de controlar altas potências.

O relê K1, além de ativar o triac, serve como isolador entre a parte de tensão mais baixa (5 V) e a de tensão mais alta (12, 110, 220 V). O conjunto C1/R2 serve para prevenir ativações falsas provenientes de cargas indutivas. F1 é um simples fusível de proteção, cuja capacidade deverá estar de acordo com a capacidade do triac O1.

**LISTA DE MATERIAIS**

C1 — Capacitor cerâmico de 0,1 uF / 50 V
D1 — Diodo 1N4002, 1N4003 ou equivalente
F1 — Fusível de ação rápida (10 A)
K1 — Relê com bobina para 5 Vcc / 50 ohms com contatos para 110 Vca / 10 A
O1 — Triac para 200 Vca / 10 A (MAC 11-4 ou equivalente)
R1 — Resitor de 1.000 ohms / 1W / 5%
R2 — Resistor de 10 ohms / 1 W / 5%

Entre as inúmeras rotinas possíveis, exemplifico algumas, que podem ser escritas em Basic, Assembly, C, dBase, Fortran ou qualquer outra linguagem que permita alterar uma saída lógica.

**Exemplo 1**

Rotina "EXECUTIVO"

06:00 h — Ligar aquecedor central
07:00 h — Acender abatjour e ligar aparelho de som
07:30 h — Imprimir agenda do dia
08:00 h — Abrir a porta elétrica da garagem
09:00 h — Desligar a chave geral
09:30 h — Ligar sistema de alarme autônomo
19:00 h — Ligar a chave geral e micro-ondas
20:00 h — Ligar a TV e a lavador de louças

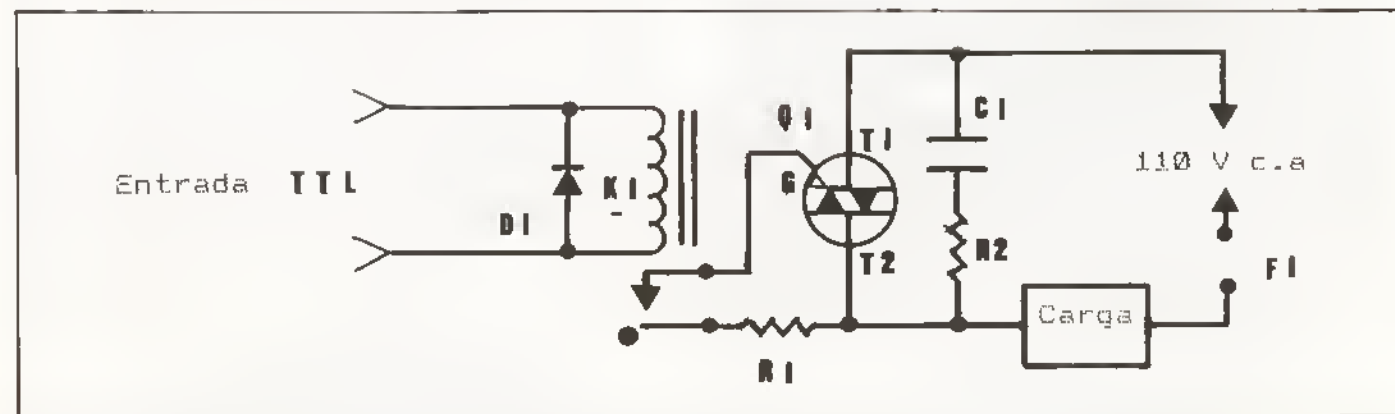
**Exemplo 2**

Rotina "SISTEMA DE SEGURANÇA"

Alguma porta ou janela aberta?
- Caso afirmativo
  - Desligar o condicionador de ar
  - Hora maior que 22 ou menor que 8?
    - Caso afirmativo
    - Soar alarme

Reiniciar rotina

O circuito abaixo foi desenvolvido partindo-se de publicações sobre interfaces de potência para micros. O assunto está longe de ser concluído, visto que cada usuário pode desejar controlar diferentes tipos de cargas em diferentes áreas e para diferentes finalidades. Para maiores informações sobre construção, projeto, aplicações ou programas, pode-se consultar a Farah's Informática, telefone 011-37.3437, São Paulo.

**Tanios Hamzo**, um dos pioneiros na microcomputação no Brasil, tanto em hardware quanto em software, é autor de livros, revistas e colunas em jornais especializados. Leciona informática e implanta cursos junto a entidades educacionais, desenvolve softwares especialistas e cria soluções eletrônicas para sistemas de hardware. Algumas delas serão publicadas futuramente.

## O MELHOR PARA SEU MSX e MSX2 VOCÊ ENCONTRA NA NEMESIS

### NEMESIS - PROGRAMAS UTILITÁRIOS

| | | |
|---|---|---|
| MSX-DOS TOOLS PLUS (LANÇAMENTO) | ferramentas para auxílio na programação | Cr$ 800,00 |
| MSX HELLO! 1.0 | multi-utilitário para uso com disk-drive | Cr$ 800,00 |
| MSX HARDCOPY 1.1 | utilitário para impressão de gráficos | Cr$ 500,00 (*) |
| EASY GRAPH | poderoso editor gráfico com recursos inéditos | Cr$ 1.200,00 |

### NEMESIS - PROGRAMAS APLICATIVOS

| | | |
|---|---|---|
| MALA DIRETA MSX 1.1 | cadastro de clientes para 7.000 registros | Cr$ 1.200,00 |
| MSX-SAM VOICE SYNTHETIZER | sintetizador de voz com 1 canal de som | Cr$ 500,00 (*) |
| MSX CHART 1.0 | gráficos comerciais e estatísticos | Cr$ 800,00 |
| MSX PORTFOLIO 1.0 | agenda eletrônica / lista telefônica | Cr$ 800,00 |
| I CHING | horóscopo chinês no computador | Cr$ 500,00 (*) |
| TEXTO TOTAL 1.0 MTA (LANÇAMENTO) | poderoso processador de textos para MTA | Cr$ 1.200,00 |
| TEXTO TOTAL 1.0 LADY (LANÇAMENTO) | poderoso processador de textos com gráficos | Cr$ 1.200,00 |
| MSX TOP CAD (LANÇAMENTO) | sensacional editor de projetos profissionais | Cr$ 1.200,00 |

### LANÇAMENTO: CONTROLE DE VÍDEO LOCADORA

Informatize seu vídeo-clube com um MSX!
Um programa super profissional de baixo custo!
Em disco com apostila completa por apenas Cr$ 2.700,00.

### NEMESIS - DESK-TOP PUBLISHING NO MSX

| | | |
|---|---|---|
| MSX PAGE MAKER 1.5 | editor de páginas com textos e gráficos | Cr$ 800,00 |
| MSX PAGE MAKER FONTES 1 | 22 diferentes letras para o PAGE MAKER | Cr$ 250,00 |
| MSX PAGE MAKER FONTES 2 | 22 diferentes letras para o PAGE MAKER | Cr$ 250,00 |
| MSX PAGE MAKER FONTES 3 | 22 diferentes letras para o PAGE MAKER | Cr$ 250,00 |
| MSX PAGE MAKER FONTES 4 | 22 diferentes letras para o PAGE MAKER | Cr$ 250,00 |
| MSX PAGE MAKER CARTOONS 1 | diversas figuras para sua página gráfica | Cr$ 250,00 |
| MSX PAGE MAKER CARTOONS 2 | diversas figuras para sua página gráfica | Cr$ 250,00 |
| MSX PAGE MAKER TITLES 1 | alfabetos gigantes para títulos e destaques | Cr$ 250,00 |
| MSX PAGE MAKER SQUARES 1 | diferentes molduras, adornos e vinhetas | Cr$ 250,00 |
| MSX PAGE MAKER KIT | PAGE MAKER com todos os seus acessórios | Cr$ 2.000,00 |

### NEMESIS CLIP-ART

UMA COLEÇÃO COM CENTENAS DE FIGURAS INÉDITAS PARA MSX PAGE MAKER OU GRAPHOS III
4 DISCOS REPLETOS DE "SHAPES" POR APENAS Cr$ 1.000,00

NOVIDADE: "NEMESIS CLIP-ART II"
MAIS FIGURAS INÉDITAS PARA MSX PAGE MAKER OU GRAPHOS III
MAIS 4 DISCOS REPLETOS DE "SHAPES" POR APENAS Cr$ 1.000,00

### NEMESIS - JOGOS E PROGRAMAS EDUCATIVOS

| | | |
|---|---|---|
| O CONDE DE MONTE CRISTO | aventura conversacional em português | Cr$ 400,00 |
| MENPHIS | aventura conversacional em português | Cr$ 400,00 |
| A GRUTA DE MAQUINÉ | aventura conversacional em português | Cr$ 400,00 |
| AUTO KIT | programa educativo para crianças | Cr$ 400,00 (*) |
| FARM KIT | programa educativo para crianças | Cr$ 400,00 |
| A TAÇA MÁGICA (SUPER LANÇAMENTO) | sensacional aventura conversacional em português | Cr$ 450,00 |

### PAULISOFT - APLICATIVOS E UTILITÁRIOS

| | | |
|---|---|---|
| SPRITE MAKER | editor de "sprites" com variados recursos | Cr$ 1.700,00 |
| MEGA MUSIC I ou MEGA MUSIC II | músicas tiradas de jogos MEGARAM japoneses | Cr$ 1.000,00 |

### APLICATIVOS E UTILITÁRIOS

| | | |
|---|---|---|
| MSX EDARQ | editor de arquivos em disco | Cr$ 1.400,00 |
| MSX VOX | digitalizador de voz 100% nacional | Cr$ 1.400,00 |
| FLUXO DE CAIXA | controle comercial de entradas e saídas | Cr$ 1.400,00 |
| CHAVE MESTRA | copiador de programas bloqueados | Cr$ 2.000,00 |
| VERSOR | copiador de programas entre 3 1/2 e 5 1/4 | Cr$ 2.000,00 |
| CADEMP | programa para cadastro de empresas | Cr$ 2.500,00 |
| CADCLI 2.0 | cadastro de clientes / mala direta | Cr$ 2.500,00 |
| NEMESIS (THE GAME) | super jogo, agora em 3 1/2 e para DD PLUS | Cr$ 1.000,00 |

### M.P.O. SOFT VÍDEO - CARTUCHOS E VÍDEOS EDUCATIVOS EM VHS

| | | |
|---|---|---|
| DOMINANDO O MSX | apresentação do MSX e seus periféricos | Cr$ 2.500,00 |
| CURSO DE BASIC 1 | iniciação e programação basic no MSX | Cr$ 4.000,00 |
| CURSO DE DBASE II | uma aula particular de DBASE II em vídeo | Cr$ 2.500,00 |
| MSX WRITE 1.0 | editor de textos em cartucho | Cr$ 1.800,00 |
| EDITOR MUSICAL | editor de música em cartucho | Cr$ 1.800,00 |
| MSX TURBO DEVICE | super acelerador para seu MSX | Cr$ 1.800,00 |
| GAME MASTER | o máximo proveito dos seus jogos (em cartucho) | Cr$ 1.800,00 |
| EDDY II | sensacional editor gráfico em cartucho | Cr$ 1.800,00 |
| MSX LOGO | a melhor LINGUAGEM LOGO em cartucho | Cr$ 2.500,00 |

### SOFT-R-MATIC - PROGRAMAS APLICATIVOS

BANCOS DE DADOS

| | | |
|---|---|---|
| MSX DATA BASE 1.1 | fichário eletrônico fácil de usar | Cr$ 400,00 (*) |
| MSX DATA BANK 1.2 | banco de dados com campos redefiníveis | Cr$ 400,00 |
| MSX EASY DATA 1.0 | cadastro redefinível fácil de usar | Cr$ 400,00 (*) |
| STOCK CONTROL 1.0 | controle de estoques fácil de usar | Cr$ 600,00 (*) |
| STOCK CONTROL 2.0 | controle de estoques profissional | Cr$ 800,00 |
| CONTAS A PAGAR/RECEBER | controle de fluxo/duplicatas e contas em geral | Cr$ 800,00 |

EDITORES GRÁFICOS

| | | |
|---|---|---|
| EDDY 1 | editor gráfico fácil de usar | Cr$ 400,00 (*) |
| EDDY 2 | editor gráfico com múltiplos recursos | Cr$ 400,00 (*) |
| CHEESE | editor gráfico fácil de usar | Cr$ 400,00 (*) |
| GRAPHIC MASTER | editor gráfico com "shapes" exclusivos | Cr$ 400,00 (*) |
| YAMAHA GRAPHIC ARTIST | editor gráfico com variados recursos | Cr$ 400,00 (*) |

| | | |
|---|---|---|
| AACKODRAW & PAINT | poderoso editor gráfico com letras e texturas | Cr$ 400,00 (*) |
| T-PAINT 1.2 | editor gráfico fácil de usar | Cr$ 400,00 (*) |
| THE DESIGNER'S PENCIL | mais que um simples editor gráfico | Cr$ 600,00 (*) |
| THE MAGIC PAINT | famoso editor gráfico do Apple agora para MSX | Cr$ 600,00 |

EDITORES MUSICAIS

| | | |
|---|---|---|
| MUSIC STUDIO G7 | poderoso editor musical com recursos inéditos | Cr$ 400,00 (*) |
| PSG MUSIC WRITER | editor de música com ritmos variados | Cr$ 400,00 (*) |
| SUPER SYNTH | poderoso sintetizador de sons | Cr$ 400,00 (*) |
| THE MUSIC EDITOR | editor de música de fácil manuseio | Cr$ 400,00 (*) |
| ELETRIC SOUND STUDIO | poderoso sintetizador de sons e efeitos musicais | Cr$ 400,00 |

PROCESSADORES DE TEXTOS

| | | |
|---|---|---|
| AACKOSCRIBE | eficiente editor de textos com 40 ou 80 colunas | Cr$ 500,00 |
| THE BANK STREET WRITER | processador de textos de fácil utilização | Cr$ 600,00 (*) |
| MSX WRITE 3.0 | a nova versão do mais famoso editor para MSX | Cr$ 500,00 |
| PRINT-X-PRESS II | editor de "desk-top publishing" para seu MSX | Cr$ 600,00 |

UTILITÁRIOS DIVERSOS

| | | |
|---|---|---|
| LETRASET 1.1 | redefinidor de caracteres com versatilidade | Cr$ 400,00 |
| SUPER COPY 7.1 | copiador de programas de fitas cassete a disco | Cr$ 400,00 (*) |
| KNIGHT COMMAND 2.0 | poderoso auxiliar para programação com MSX | Cr$ 400,00 (*) |
| D.D.S. HELP | auxilia no uso do MSX com "disk-drive" | Cr$ 400,00 |
| DISK-IT | facilita a operação do MSX com "disk-drive" | Cr$ 600,00 |
| MSX DUAD 7.4 | montador assembler / desassembler, editor, etc. | Cr$ 500,00 |

PROGRAMAS DE USO GERAL

| | | |
|---|---|---|
| MSX GAME DESIGNER | faça os seus próprios jogos com facilidade | Cr$ 500,00 (*) |
| STAR SEEKER | excelente programa para amantes de astronomia | Cr$ 400,00 (*) |
| PRINT SEXY SHOP | imprime mulheres nuas com sua impressora | Cr$ 400,00 |
| PSYCHEDELIA | impressionantes efeitos gráficos no seu MSX | Cr$ 400,00 |

APLICATIVOS PROFISSIONAIS PARA MSX2 64KB

| | | |
|---|---|---|
| PHILLIPS VIDEO GRAPHICS | impressionante editor gráfico com manual | Cr$ 800,00 |
| FREE HAND | excelente editor gráfico alta-resolução com CAD | Cr$ 500,00 |
| DYNAMIC PUBLISHER | programa de desk-top para o MSX2 com manual | Cr$ 800,00 |

APLICATIVOS PROFISSIONAIS PARA MSX2 128 KB ou MEMORY MAPPER

| | | |
|---|---|---|
| PHILLIPS EASE 3.0 | integrado PLANILHA/DESK-TOP/DATABASE/CHARTS c/manual Cr$ 1.800,00 | |
| MSX2 EXCLUSIVE NORTON UTILITIES | programa de testes e recuperação de disketes para MSX2 | Cr$ 1.000,00 |

### VÍDEO-GAMES MSX 1

CYBERBIG, BLACK TIGER 1, BLACK TIGER 2, BLOODY, CAPTAIN TRUENO 1, CAPTAIN TRUENO 2, TOY ACID GAME, COSMIC SHERIFF, MIKE GUNNER (LASER PISTOL), LIVINGSTONE II PART 1, LIVINGSTONE II PART 2, MECANO OASIS, XH-63 PATROL, ZANAC III, TENSION, WARLORDS, RANSACK, TALEF, HYPERTRONIC, RATH-AE, SWING (NOVIDADE DA COMPILE), SABOTAGE, PASTEMAN e THE SMURFS, MORTADELO & SALAMINHO II, BUMPY CAT, WAR IN MIDDLE EARTH, SONIC ARKANOID, CHASE HQ, AZTEC ADVENTURE, ELETRIC CAT LAND, MEGANOVA, RAM (THE WAR), MOTORBIKE MADNESS.

BATMAN (THE MOVIE), SHANGAI SURPRISE, GHOSTBUSTERS II, VICARIO TENNIS, SHINOBI, GEMINI WING, WINTER GAMES EDITION e INTERNATIONAL BASKET (LANÇAMENTOS):

CADA JOGO POR Cr$ 50,00, 5 JOGOS POR Cr$ 200,00 ou 10 JOGOS POR Cr$ 300,00
RAMBO III em disco: apenas Cr$ 300,00
MASK II em disco: apenas Cr$ 300,00

### PROGRAMAS MEGARAM MSX1:

KING KNIGHT, JAGUR, SAMURAI KONAMI, KING'S VALLEY II, CRAZY, FANTAZY ZONE, VAXOL, OOLVELLIUS, ETC...

CADA UM EM DISCO POR APENAS Cr$ 300,00

### PROGRAMAS MEGARAM MSX2:

R-TYPE, YAM, DYNAMIT BOWL, ZOMBIE HUNTER, KING'S VALLEY II, DRAGON BUSTERS, BASEBALL II KONAMI, DRUID, DEEP FOREST, WAR OF THE DEAD, SPACE MANBOW, TOPPLE ZIP II, EAGLE WAR, DIRES, ASHGUINE, STAR SHIP WAR, HARDBALL, OUT RUN, ETC...

CADA UM EM DISCO POR APENAS Cr$ 300,00

### JOGOS MEGARAM SEM MEGARAM (MSX1 NORMAL):

NEMESIS, VAXOL, MIRAI, SUPER LAYDOCK e FINAL ZONE.

CADA UM EM DISCO POR APENAS Cr$ 500,00

### - ATENÇÃO -

Garantimos aos nossos clientes 5 anos de assistência para os produtos que comercializamos. Os programas da NEMESIS INFORMÁTICA são de origem 100% nacional, registrados pela própria empresa ou por seus autores. Não nos responsabilizamos por programas de empresas e que representamos;

Ao comprar esses programas em empresas revendedoras autorizadas, confira na embalagem se o produto é original. Não deixe que o pirata lhe engane;

Os programas acima estão disponíveis em 5 1/4 e 3 1/2. Para 3 1/2 acrescente Cr$ 200,00 por programa;
O utilitário MSX HELLO! 1.0 não está disponível em 3 1/2;
Os programas assinalados com asterisco (*) podem ser gravados em FITA CASSETE;
Consulte-nos sobre gravações de jogos em FITA CASSETE;
Clube do leitor: 10% em todas as compras;
O pedido mínimo é de Cr$ 1.000,00;
Esta tabela está válida até o final de nossos estoques.

ENVIE VALE POSTAL OU CHEQUE NOMINAL A NEMESIS INFORMÁTICA LTDA.
CAIXA POSTAL 4583 CEP. 20001 RIO DE JANEIRO - RJ OU VENHA PESSOALMENTE AO SHOW-ROOM NEMESIS
RUA SETE DE SETEMBRO 92 SALA 1910 - CENTRO - RIO DE JANEIRO - RJ - TELEFONE: (021) 222-4900

# NAS MALHAS DO COBOL

CARLOS ALBERTO HERSZTERG

Ouase nada se escreveu até hoje sobre o Cobol para MSX. E o pouco que apareceu se resume a modestas menções em listas de linguagens disponíveis para essa máquina. Diante desse vazio de informações, o presente artigo visa mostrar todas as particularidades do compilador Cobol mais popular (e mais completo) para as máquinas de 8 bits — o Microsoft Cobol-80.

Neste número de CPU serão apresentadas rotinas de interfaceamento estritamente técnicas, destinadas aos já conhecedores do Cobol-80 e com alguma experiência em Assembly. Abordaremos um aspecto obscuro desse compilador: sua total customização para o MSX. Ouem já tentou fazê-lo, sabe como é árdua a tarefa. Entretanto, mostraremos, de forma clara e objetiva, como adaptar o Cobol para trabalhar em telas de 40 ou 80 colunas, como utilizar as setas, como programar as teclas de função e muito mais.

Essas adaptações têm por objetivo principal reunir todas as potencialidades do Cobol com os recursos do MSX, e também estreitar os laços de compatibilidade e transportabilidade de software entre o Cobol-80 e o MS-Cobol (para IBM-PC).

Além dessas adaptações, o leitor também encontrará, nestas páginas, valiosas informações do uso desse compilador, sendo que muitas delas sequer aparecem no manual original.

#### O COBOL-80

O pacote do compilador Cobol-80 contém, originalmente, os seguintes arquivos:

| Compilação | Link/Execução | Software utilitário |
|---|---|---|
| COBOL.COM | COBLOC | L80.COM |
| COBOL1.OVR | COBLBX.REL | LIB.COM |
| COBOL2.OVR | CRTDRV.REL | M80.COM |
| COBOL3.OVR | SORTLIB.REL | REE80.COM |
| COBOL4.OVR | DEBUG.REL | |
| | RUNCOB.COM | |

A configuração acima pertence ao Cobol-80 na versão 4.60, a qual me parece ser a mais recente.

Além desses arquivos, existem muitas bibliotecas que são facilmente encontradas. Entre elas as mais interessantes são:

ISAMX .REL (múltiplas chaves para arquivos indexados)
LIVRO1 .REL (diferenças de datas, calendários perpétuos
LIVRO2 .REL reindexação de arquivos avariados, criptografia de dados, compactação de arquivos, entre outras facilidades)
LIVROn .REL
COBMATH .REL (rotinas matemáticas escritas em Fortran)
FINANT .REL (rotinas financeiras)

Para evitar eventuais desencontros com os leitores de CPU, saliento que perdi todas essas bibliotecas por ocasião de uma operação mal sucedida. Ouem tiver algumas delas (ou outras) e necessitar de orientações, me coloco à disposição para esclarecimentos.

#### A CUSTOMIZAÇÃO DO COBOL-80 PROCEDIMENTOS INICIAIS

Antes de mais nada, tire uma cópia back-up do pacote do Cobol-80. Mesmo que você já possua o back-up citado, tire uma nova cópia. Uma vez realizada a referida cópia de segurança, pegue um disco vazio e copie para ele os seguintes arquivos:

1 — Algum DOS (MSX-DOS, SOLX-DOS, etc)
2 — Oualquer editor de texto, exceto aqueles baseados no TassWord (MSX-Word, MSX-Text, etc). O editor do Turbo Pascal é o ideal, pois permite a execução de programas transientes sem ter que voltar ao DOS.
3 — O Macro Assembler M80.COM
4 — O Link-Loader L80.COM
5 — O Library Manager LIB.COM
6 — O CRT Driver denominado CRTDRV.REL
7 — A biblioteca standard do Cobol COBLBX.REL ( ou COBLIB.REL, dependendo da versão).

Esse disco, recém-criado, será nosso disco de trabalho no processo de customização do Cobol-80.

#### AS ROTINAS DE VÍDEO E TECLADO

O Cobol-80 possui um driver de vídeo e teclado chamado CRTDRV.REL que, por ser realocável, não permite alterações diretas. Nesse arquivo estão definidas todas as constantes de teclas, parâmetros como as dimensões da tela e rotinas gerais de CRT I/O. Sempre que há um Display ou Accept no programa Cobol, esse arquivo é procurado por ocasião da link-edição.

Como já disse, o arquivo não permite alterações: qualquer erro temos que reescrevê-lo. Com o programa fonte criado é fácil realizar qualquer alteração posterior, baslando, para isto, editar o módulo CRTDRV e, em seguida, o recompilar.

Para um perfeito entendimento dos programas listados neste artigo, vamos definir alguns conceitos:

Módulo é o nome que se dá a todo o conjunto de rotinas ou constantes definidas como Entry Points. Um módulo é, portanto, um conjunto de rotinas relacionadas a algum procedimento — como, por exemplo, rotinas de acesso ao disco, rotinas matemáticas e, como na figura 1, rotinas de manipulação do console (CRT).

Um Entry Point é um endereço realocável, de valor relativo dentro do módulo.

Já uma Biblioteca, no sentido em que venho me referindo, é um conjunto de módulos agregados com um só nome físico (como COBLBX.REL).

Isso posto, acompanhe a listagem do módulo CDMSX (figura 1) e identifique seus Entry Points e outros elementos como se segue:

**TITLE CDMSX** — Usualmente, o título dá o nome lógico ao módulo

**ENTRY's vários** — Define quais Entry Points o módulo possui

(16 Entry Points neste caso).

**EXT** — Define que os rótulos posteriores à declaração EXT constarão no módulo como chamadas externas, ou seja, estão definidos em outro lugar. Durante a link-edição, esses endereços são pesquisados e resolvidos.

**$CRWID** — Constante que define o número de colunas

**$CRLEN** — Constante que define o número de linhas

**$CLIST** — Lista de teclas usadas na edição de campos. Note que não é possível a inserção de caracteres.

**$TLIST** — Lista de teclas de finalização do formato 3 do Accept (Accept tela). Repare que seta acima e seta abaixo permitem que se "passeie" pela tela definida na Screen Section.

**$FLIST** — Lista de teclas que finalizam o último ACCEPT tela (formato 3). O código indicado como ESC Code é aquele obtido por meio da sentença Accept Item From Escape Key.

Ao lado das trÊs listas acima, estão relacionadas as teclas associadas aos códigos definidos: o padrão ANSI (American National Standards Institute), relacionado àquelas funções, que figura em toda a literatura de Cobol; a função da tecla; e, no caso de $TLIST e $FLIST, o código de escape produzido pela tecla.

Em $FLIST, a menção das teclas de função de F1 a F10, entre parênteses, define a correlação que será implementada na figura 2.

Todas as listas são finalizadas pelo byte 0.

Se vocÊ desejar redefinir teclas, tome o cuidado de não repeti-las. Isto significa que uma tecla só pode aparecer uma vez nas três listas.

$SETCR — Envia a sequência para o posicionamento do cursor (padrão DEC VT52). Observe que Lin e Col são somados com 31 (1FH) e não com 32 (20H), pois, para o Cobol, o topo da tela tem as coordenadas 1,1.

$CURON e $CUROF — Enviam as sequências de liga/desliga o cursor. Entretanto, para usufruir dessas características, o envio de caracteres à tela é incrivelmente retardado. Se você não se incomodar em ter o cursor sempre presente, faça:

```
$CURON: (label vazio)
$CUROF: RET
```

Isto deverá acelerar a impressão na tela.

$HILIT e $LOLIT — Ativa e desativa o vídeo reverso (ou saída em alto brilho). Não existe no MSX, mas podem ser criadas rotinas de inversão e colocadas nestes Entry Points.

$CURBK — Move o cursor para trás, apagando o caracter à esquerda. Tecla <BS>.

$ERASE — Envia a sequência de apagamento parcial da tela. Apaga da posição do cursor até o fim da tela.

$EOL — Envia a sequência de apagamento parcial da linha. Apaga da posição do cursor até o fim da linha.

$ALARM — Envia o caracter Bell (07 ASCII), emitindo um Beep quando requisitado em um Accept.

$INCRT — Entrada de caracteres. Pode conter rotinas de filtragem de caracteres, bastando para isto fazer:

```
$INCRT: CALL $INKEY
.
. (sua rotina)
RET
```

$OUCRT — Saída de caracteres.

Uma vez estudado todo o corpo do CRT Driver, vamos implementá-lo. Creio que, para os conhecedores de Assembly, as rotinas estão claras o suficiente para possíveis modificações, sendo a alteração do valor da constante $CRWID para 80 a modificação mais imediata, caso você possua uma placa de 80 colunas ou o MSX-2.

Para criar o Driver, siga os seguintes passos:

1 — Edite o programa listado na figura 1 e salve-o com o nome CDMSX.MAC,

2 — Compile o módulo, comandando do DOS o M80 do seguinte modo: M80 CDMSX=CDMSX

Pronto. Agora o arquivo CDMSX.REL contém as rotinas específicas do tratamento do console do MSX. Para facilitar ainda mais as coisas, vamos associar as teclas de função aos códigos de escape (definidos em $FLIST) de 02 a 11. Para tanto, incluiremos mais um módulo na biblioteca de link-edição COBLBX.REL.

A figura 2 mostra um programa simples que define dois Entry Points: um que reconfigura as teclas de função, salvando as anteriores (COBKEY) e outro que reestabelece as teclas de função previamente salvas (MSXKEY).

Por etapas:

1 — Edite o programa listado na figura 2 salvando-o com o nome KEYS.MAC.

2 — Compile o novo módulo, comandando no DOS: M80 KEYS=KEYS,

3 — Comande no DOS: LIB. Neste ponto, toda atenção é pouca. Estamos agora trabalhando com o gerenciador de bibliotecas do pacote de software utilitário da Microsoft. Este programa, quando mal utilizado, pode destruir irreversivelmente a biblioteca em uso. Para evitar confusões, lembro, mais uma vez, que a biblioteca standard do Cobol pode se chamar COBLBX.REL ou COBLIB.REL, dependendo da sua versão do compilador Cobol.

Já dentro do LIB, o prompt '*' aparecerá, aguardando comandos. Digite, então:

INTLIB=COBLBX,KEYS

Após algum tempo de trabalho em disco, o prompt '*' reaparecerá. Neste ponto, digite /E para finalizar a seção de trabalho com o LIB e voltar ao DOS. Como resultado, uma nova biblioteca deve ter sido criada com o nome de INTLIB.REL, mantendo a anterior (COBLBX.REL) intacta.

Verifique no diretório se o arquivo INTLIB.REL foi realmente criado e compare o tamanho deste com o tamanho da biblioteca anterior. A diferença entre ambos deve ser superior a 200 bytes e inferior a 400 bytes.

### ROTINAS DE CAPTURA DE DADOS DO DOS

O formato 1 do comando Accept permite que se obtenham várias informações. São elas: o valor do último escape code, o número do terminal em uso, a hora do sistema, a data juliana do sistema e a data normal do sistema. Entretanto, dada a diversidade de equipamentos que rodam o Cobol-80 sob o CP/M 2.2, é possível que a sua biblioteca standard contenha rotinas específicas para obter dados do sistema da máquina onde o seu compilador foi transcrito (Apple, TRS-80, EBC-4010/4020, etc). Possivelmente, por isso, a data, a data juliana e hora sempre aparecem erradas quando requisitadas em um Accept.

Com relação ao escape code e ao número do terminal (linha), não existem problemas, pois o primeiro é manipulado internamente pelo próprio Cobol em tempo de execução. Já o LINE NUMBER especifica o terminal em uso e não faz sentido em sistemas monousuários, retornando sempre com o valor 00 quando requisitado.

A figura 3 mostra a listagem do módulo ACPDAT, que contém todos os Entry Points do módulo original, a ser substituído na biblioteca INTLIB.REL (a versão atualizada da biblioteca original). Estude com atenção a listagem do módulo ACPDAT observando os comentários.

Uma única observação se faz necessária com relação à hora do sistema, que só faz sentido em sistemas que dispõem de um circuito de relógio interno, como o MSX-2. Por falta de informações a esse respeito, o EQUATE HORSIS aponta para uma região da memória onde só aparecem zeros. Entretanto, quando colocado neste EQUATE o endereço real, onde é armazenada a hora do MSX-2, se processará a contento.

Entendida a lógica geral do módulo, vamos implementá-lo, observando os seguintes passos:

1 — Edite o módulo ACPDAT e salve-o com o nome ACPDAT.MAC.

2 — Compile o módulo comando do DOS: M80 ACPDAT=ACPDAT.

3 — Carregue o LIB e quando o prompt '*' aparecer, digite: MSXLIB=INTLIB<..ACPDAT-1>,ACPDAT

Após algum tempo, o LIB voltará ao modo comando, exibindo o prompt '*'. Comande, então:

INTLIB<ACPDAT+1..>/E

Agora todas as implementações estão contidas no arquivo MSXLIB.REL. Se você reparou com atenção todos os passos que seguiu até aqui, evidentemente sabe que o arquivo MSXLIB.REL é a nova biblioteca COBLBX.REL e que o arquivo CDMSX.REL é o novo CRTDRV.REL. Copie esses arquivos para o disco back-up que você tirou, renomeando-os, isto é, comandando no DOS:

COPY CDMSX.REL B:CRTDRV.REL /V
COPY MSXLIB.REL B:COBLBX.REL /V (ou B:COBLIB.REL)

### TESTANDO A NOVA CONFIGURAÇÃO

A figura 4 mostra um programa Cobol que efetua todos os testes das rotinas implementadas. O programa é auto-explicativo e, além de realizar os testes mencionados, contém alguns truques como, por exemplo, representar caracteres fora da faixa de 32 a 127 ASCII.

### GUIA DE COMANDOS E CHAVES DO COMPILADOR E DO LINK

Compilador Cobol — linha de comando
1) Compilação normal
COBOL [nomeobj]=nomeprog
2) Compilação com listagem na tela
COBOL [nomeobj],TTY:=nomeprog
3) Compilação com listagem na impressora
COBOL [nomeobj],LST:=nomeprog
4) Compilação com criação de arquivo-listagem
COBOL [nomeobj],nomelist=nomeprog

Não utilize extensões no nome do arquivo-objeto (nomeobj) nem no nome do arquivo-fonte (nomeprog) que, por default, são: REL e COB, respectivamente.

### COMPILADOR COBOL — CHAVES

/D — Evita a criação do arquivo "nomedebug".DBG, onde "nomedebug" representa os seis primeiros caracteres do nome

- os melhores cursos -

assistência técnica especializada -

- mais de 40.000 clientes -

- o maior estoque do mercado -

- mais de 2.000 programas -

a mais completa linha de periféricos -

Equipamentos · Acessórios · Periféricos

Interfaces · Drives · 80 colunas · Modem

**O MAIOR SHOW ROOM DO PAÍS !!!**

RUA APIACÁS, 92 – SÃO PAULO - CEP 05017 / FONE 872-0730

```
TITLE CIMSX - MSX COBOL CRT DRIVER

ENTRY $ALARM,$CLIST,$CRLEN,$CRWID
ENTRY $CLABK,$CUROF,$CURON,$EOL
ENTRY $ERASE,$FLIST,$HILIT,$INCRT
ENTRY $LOLIT,$OUCRT,$SETCR,$TLIST

EXT   $OUTCH,$INKEY

        .Z80

$CRWID: DB 40          ; Número de colunas
$CRLEN: DB 24          ; Número de linhas
                       ;------------------------------------------------;
                       ;              Teclas de edição de campos        ;
                       ;---------------+-------------+------------------;
                       ;    Tecla      | Padrão ANSI |      Função      ;
                       ;---------------+-------------+------------------;
$CLIST: DB 15H         ; CONTROL-X     | CONTROL-U   | Apaga linha      ;
        DB 08H         ; BS            | DELETE      | Apaga caracter   ;
        DB 1CH         ; SETA DIREITA  | CONTROL-F   | Cursor para frente ;
        DB 1DH         ; SETA ESQUERDA | CONTROL-H   | Cursor para trás ;
        DB 2BH         ; +             | (2BH ASCII) | Sinal positivo   ;
        DB 2DH         ; -             | (2DH ASCII) | Sinal negativo   ;
        DB 0           ; Fim da lista  |             |                  ;
                       ;---------------+-------------+------------------;

                       ;--------------------------------------------------------;
                       ;        Teclas de término da instrução ACCEPT           ;
                       ;---------------+-------------+------------------+-------;
                       ;    Tecla      | Padrão ANSI |      Função      |Esc code;
                       ;---------------+-------------+------------------+-------;
$TLIST: DB 1EH         ; SETA ACIMA    | CONTROL-B   | Campo anterior   |  99  ;
        DB 1BH         ; ESCAPE        | ESCAPE      | ACCEPT escape    |  01  ;
        DB 1FH         ; SETA ABAIXO   | CONTROL-T   | Campo posterior  |  09  ;
        DB 0DH         ; RETURN        | RETURN      | Término normal   |  00  ;
        DB 0AH         ; LF            | LF          | Término normal   |  00  ;
        DB 0           ; Fim da lista  |             |                  |      ;
                       ;---------------+-------------+------------------+-------;

                       ;--------------------------------------;
                       ;       Teclas de ESCAPE do ACCEPT     ;
                       ;---------------+-------------+--------;
                       ;    Tecla      | Padrão ANSI |Esc code;
                       ;---------------+-------------+--------;
$FLIST: DB 01H         ;CONTROL-A (F1) | CONTROL-A   |  02   ;
        DB 02H         ;CONTROL-B (F2) | CONTROL-C   |  03   ;
        DB 03H         ;CONTROL-C (F3) | CONTROL-K   |  04   ;
        DB 04H         ;CONTROL-D (F4) |   --        |  05   ;
        DB 05H         ;CONTROL-E (F5) |   --        |  06   ;
        DB 06H         ;CONTROL-F (F6) |   --        |  07   ;
        DB 07H         ;CONTROL-G (F7) |   --        |  08   ;
        DB 0BH         ;CONTROL-K (F8) |   --        |  09   ;
        DB 0CH         ;CONTROL-L (F9) |   --        |  10   ;
        DB 0EH         ;CONTROL-N (F10)|   --        |  11   ;
        DB 0           ; Fim da lista  |             |       ;
                       ;---------------+-------------+--------;

$SETCR: LD   A,1BH    ; Rotina de posicionamento do cursor
        CALL $OUTCH   ; (sequencia ESC Y Lin+31 Col+31)
        LD   A,59H
        CALL $OUTCH
        LD   A,H
        ADD  A,1FH
        CALL $OUTCH
        LD   A,L
        ADD  A,1FH
        JP   $OUTCH

$CURON: LD   A,1BH    ; Habilita o cursor (sequencia ESC x 5).
        CALL $OUTCH
        LD   A,78H
        CALL $OUTCH
        LD   A,35H
        JP   $OUTCH

$CUROF: LD   A,1BH    ; Apaga o cursor (sequencia ESC y 5).
        CALL $OUTCH
        LD   A,79H
        CALL $OUTCH
        LD   A,35H
        JP   $OUTCH

$HILIT:               ; Atributos de vídeo não disponiveis no MSX.
$LOLIT: RET

$CLABK: LD   A,08H    ; Move cursor para trás.
        JP   $OUTCH

$ERASE: LD   A,1BH    ; Apaga da posição do cursor até o fim da TELA.
        CALL $OUTCH
        LD   A,4AH
        JP   $OUTCH

$EOL:   LD   A,1BH    ; Apaga da posição do cursor até o fim da LINHA.
        CALL $OUTCH
        LD   A,4BH
        JP   $OUTCH

$ALARM: LD   A,07H    ; Bell (BEEP)
        JP   $OUTCH

$INCRT: JP   $INKEY   ; Entrada de caracteres
$OUCRT: JP   $OUTCH   ; Saida de caracteres

END
```

FIGURA 1

KONAMI SOFTWARE

ATENÇÃO
COMUNICAMOS AOS CLIENTES E AMIGOS QUE A SOCIEDADE DENOMINADA KONAMI SOFTWARE LTDA. PASSA A PARTIR DESTE MOMENTO A CHAMAR-SE TAKERU SOFTWARE INFORMÁTICA LTDA., DANDO CONTINUIDADE AOS BONS SERVIÇOS PRESTADOS ATÉ O MOMENTO.

# TAKERU SOFTWARE

MSX 1 MSX 2 MSX 2+

## MSX 1

*CORSÁRIOS II*
*WAR IN MIDDLE EARTH*
*R.A.M.*
*MEGANOVA*
*CAT & WOLF*
*FIVE BALLS*
*MORTADELA & FILEMON II – 1*
*MORTADELA & FILEMON II – 2*
*DIOSA DE COZUMEL I*
*DIOSA DE COZUMEL II*

*Cr$ 80,00 CADA JOGO (DISCO NÃO INCLUÍDO)*

## DIGITALIZAÇÃO

AGORA VOCÊ JÁ PODE OUVIR EM SUA CASA AS TRILHAS SONORAS ORIGINAIS (ORCHERSTRADS & VOICES) DOS JOGOS PREFERIDOS GRAVADOS DE C.D. PARA FITAS CASSETE.
LANÇAMENTOS DO MÊS:
* VALIS II * R-TYPE II
* BURAI * X MULTIPL
* GRADIUS FANTASIA
* GRADIUS II (GOFER)
* GRADIUS (BATTLE MUSIC COLLECTION)
* SALAMANDER (BATTLE MUSIC COLLECTION)
* Cr$ 700,00 CADA TRILHA SONORA (FITA INCLUSA)

## MSX 1

*JOGOS ADAPTADOS RODANDO SEM O CARTUCHO MEGARAM*

* *SUPER LAYDOCK*
* *VAXOL*
* *FINAL ZONE*
* *MIRAI*
* *FANTASY ZONE*

*Cr$ 400,00 CADA JOGO (DISCO 5 1/4 INCLUSO)*

## MSX 2 · 2DD

*REVIVER – AVENTURA RPG – 1 X 2DD*
*TWINKLE STAR – ESPACIAL – 1 X 2DD*
*PSY-O-BLADE – ADVENTURE ANIMADO – 3 X 2DD*
*DRAGON WORLD – ADVENTURE – 1 X 2DD*
*SPEED GAME – ERÓTICO – 1 X 2DD*
*RUNE WORTH – AVENTURA RPG – 4 X 2DD*
*RENDEZVOUS AVEC RAMA – ADVENTURE – 2 X 2DD*
*ZOO – ADVENTURE – 2 X 2DD*

*Cr$ 150,00 POR DISCO – DISCO NÃO INCLUSO*

## MEGAROM

*QUARTH – RACIOCÍNIO – MEGARAM MSX 2.0*
*QUARTH – RACIOCÍNIO – MEMORY MAPPER MSX 2.0*
*PREDATOR – GUERRA – MEMORY MAPPER MSX 2.0*
*MR GHOST – AVENTURA – MEMORY MAPPER MSX 2.0*
*METAL GEAR (INGLÊS) – ESTRATÉGIA – MEMORY MAPPER MSX 2.0*
*LASER SQUAD – ESTRATÉGIA – MEMORY MAPPER MSX 2.0*
*ANIMAL LAND – ADVENTURE – MEMORY MAPPER MSX 2.0*
*HYDLIDE 3 – AVENTURA RPG – MEMORY MAPPER MSX 2.0 – 2DD*
*COSMOS CLUB – ADVENTURE – MEMORY MAPPER MSX 2.0 – 2DD*
Cr$ 200,00 CADA JOGO – DISCO NÃO INCLUSO

## PERIFÉRICOS

COM O MENOR PREÇO E UM ANO DE GARANTIA
* IMPRESSORAS – EM PROMOÇÃO
* DRIVES 5 1/4 & 3 1/2 – EM PROMOÇÃO
* KIT MSX 2.0 – EM PROMOÇÃO
* MEGA RAM – EM PROMOÇÃO
* MEGA RAM DISK – EM PROMOÇÃO
* CARTUCHO DE 80 COLUNAS
* EXPANSOR DE SLOTS
* COMPUTADORES
* JOYSTICKS
* MODEM
* MONITORES
* MESAS P/ COMPUTADORES/ IMPRESSORAS
* SUPORTE PARA IMPRESSORA
* ESTABILIZADORES DE CORRENTE
* FORMULÁRIOS CONTÍNUOS
* ETIQUETAS
* DISQUETES 5 1/4 E 3 1/2
* PORTA-DISQUETES
* CAIXA DE MADEIRA C/ TAMPA P/ 3 1/2 E 5 1/4
* PREÇOS SOB CONSULTA

REVENDA AUTORIZADA **MSX-SOFT INFORMÁTICA.**

## APLICATIVOS - DEMOS - UTILITÁRIOS

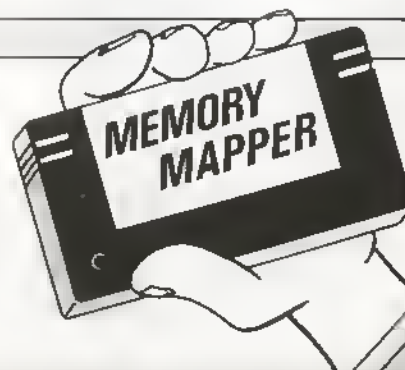
MSX 2 MSX 2+

* *CPM 3.0* Cr$ 2.000,00 DISCO INCLUSO — O melhor CPM do momento com várias funções implementadas (criar subdiretórios, alterações de cores, etc.). Versão em 3 1/2 & 5 1/4 (720 Kbytes). Necessita do cartucho Memory Mapper.
* *HSH D.O.S.* Cr$ 1.800,00 DISCO INCLUSO — Um super D.O.S. com várias funções novas (color, ramdisk, keydos, etc.). Versão em 5 1/4 & 3 1/2. Necessita do cartucho Memory Mapper.
* *DANTE* Cr$ 1.800,00 DISCO INCLUSO — Um sensacional editor de games RPG. Agora você vai poder editar o seu próprio jogo. Confira! Versão em 3 1/2 & 5 1/4 (720 Kbytes). Necessita do cartucho Memory Mapper.
* *LIGHTING DEMO* Cr$ 600,00 DISCO INCLUSO — Demo gráfico – Versão 3 1/2 & 5 1/4 (720 Kbytes). Necessita do cartucho Memory Mapper.
* *STAR WARS DEMO* Cr$ 600,00 DISCO INCLUSO — Demo gráfico – Versão 3 1/2 & 5 1/4 (720 Kbytes). Necessita do cartucho Memory Mapper.

## REVENDA AUTORIZADA

*SCREEN STEALER* – 25 BTNs
*POSTER MAKER* – 53 BTNs
*PROFESSIONAL PUBLISHER* – 123 BTNs
*FLOW CHART* – 53 BTNs

## MEMORY MAPPER

Você que não adquiriu a sua memory mapper não perca tempo ligue hoje mesmo para Takeru. E passe a utilizar os sensacionais lançamentos acima.
Maiores informações pelo tel. (021) 232-0650.

Pedido mínimo: Cr$ 800,00
Disquete 5 1/4: Cr$ 90,00 – Disquete 3 1/2: Cr$ 350,00
Para adquirir os nossos produtos basta nos enviar o seu pedido em uma folha com nome, endereço, etc., juntamente com o cheque a TAKERU INFORMÁTICA LTDA. – Rua Sete de Setembro, 92 – Sala 2210 – Centro – Rio de Janeiro – CEP 20050 – Tel. (021) 232-0650.
*Remetemos os nossos produtos para todo o Brasil via Sedex.*
*Solicite hoje mesmo o nosso super catálogo inteiramente grátis.*

DSC

do programa declarado no PROGRAM-ID. Esta chave, quando presente, além de aumentar as velocidades de compilação, link-edição e execução, gasta menos memória e menos espaço em disco. Em contrapartida, não fornece o número da linha onde ocorreu algum erro em tempo de execução.

/C — Força o compilador a criar o arquivo "nomeprog".CRF para efeito de geração de referências cruzadas pelo CREF80.COM.

/Fn — Mostra, sob forma de advertência (Warning) todos os comandos acima de FIPS n (Federal Information Processing Standard level n) onde 'n' é um digito entre 0 e 4.

/F0 — Assinala tudo acima de LOW LEVEL
/F1 — Assinala tudo acima de LOW INTERMEDIATE LEVEL
/F2 — Assinala tudo acima de HIGH INTERMEDIATE LEVEL
/F3 — Assinala tudo acima de HIGH LEVEL
/F4 — Não assinala nada (default)

/L — Força o compilador a gerar um arquivo-listagem de nome "nomeprog".PRN.

/P — Espaço para a pilha. Se, durante a compilação, surgir a mensagem "? Stack overflow", use /P para dispor mais espaço à pilha. Cada /P colocado aloca 100 bytes extras para a pilha.

/R — Força o compilador a gerar o arquivo-objeto, com o mesmo nome do fonte e no mesmo disco que este, com o nome "nomeprog".REL. Use em caso de problemas na geração do objeto ou se for omitido o nome desse na linha de comando.

**L80 Link-Loader — linha de comando**

1) Link para programas sem chamadas externas
L80 nomeobj,nomecom / chaves
2) Link para programas com chamadas externas
L80 nomeobj-1,nomeobj-2,...,nomeobj-n,nomecom / chaves

Observação: nomeobj são nomes de arquivos relocáveis (já compilados com extensão .REL default)

nomecom é o nome que deve ser dado ao programa executável (extensão .COM default)

**L80 Link-Loader — chaves**

O L80 possui, ao todo, 12 chaves. Entretanto, para link-edição de programas Cobol, usualmente apenas três são usadas, aos pares:

/N/E — Gera o arquivo executável "nomecom".COM e volta ao sistema operacional.

/N/G — Gera o arquivo executável "nomecom".COM e inicia a execução do mesmo, automaticamente.

Com base nestas informações, a figura 5 mostra alguns arquivos-de-lote (.BAT) que podem ser criados para facilitar a compilação e link-edição de seus programas.

## CONCLUSÃO

Em se tratando de Cobol, tudo o que se escreveu terá sido pouco. Neste artigo todos os velhos problemas de customização foram resolvidos (espero!) e algumas técnicas foram mostradas. Dessa forma, devo ter ajudado todos os programadores Cobol, profissionais ou estudantes, no sentido de aproveitarem ao máximo as facilidades do MSX na programação dessa linguagem.

Obviamente, o assunto Cobol não terminará aqui. Oportunamente veremos, nas páginas de CPU, outras abordagens acerca dessa linguagem, tais como a criação de sub-rotinas escritas em Cobol e Assembly, criação de bibliotecas, etc.

Com relação ao material aqui apresentado, me coloco à inteira disposição dos leitores de CPU, em caso de dúvidas.

■ **Carlos Alberto Herszterg**, formado em programação pelo Instituto Brasileiro de Pesquisa em Informática (IBPI — RJ), é estudante do curso de Tecnólogo em Processamento de Dados da Pontifícia Universidade Católica do Rio de Janeiro (PUC - RJ) e programador em Cobol, Basic, Pascal, dBase e Assembly.

```
TITLE KEYS

;; Modos de chamadas:
;;   CALL "COBKEY" - Programa as teclas de função e salva as anteriores
;;   CALL "MSXKEY" - Repõem as teclas previamente salvas

ENTRY   COBKEY,MSXKEY

        .Z80

ZERO15  MACRO
          REPT 15         ;Gera 15 definições DB 0
            DB 0
          ENDM            ;Fim da repetição
        ENDM              ;Fim da Macro

FNKSTR  EQU  0F87FH       ;Endereço das Teclas de função

COBKEY: EXX               ;Salva todos os registradores
        EX   AF,AF'
        LD   HL,FNKSTR    ;Salva o conteúdo das teclas
        LD   DE,KEYBUF
        LD   BC,0A0H
        LDIR
        LD   HL,KEYCOB    ;Coloca as definições de teclas
        LD   DE,FNKSTR    ;na área de texto das teclas de função
        LD   BC,0A0H
        LDIR
        EXX               ;Repoem todos os registradores
        EX   AF,AF'
        RET

MSXKEY: EXX               ;Salva todos os registradores
        EX   AF,AF'
        LD   HL,KEYBUF    ;Repoem as teclas salvas em KEYBUF
        LD   DE,FNKSTR
        LD   BC,0A0H
        LDIR
        EXX               ;Repoem todos os registradores
        EX   AF,AF'
        RET

        DSEG              ;Segmento de dados

KEYBUF: DS 160

KEYCOB: IRP X,<1,2,3,4,5,6,7>
          DB  X           ;F1 a F7 - Control-A a Control-G
          ZERO15
        ENDM

        DB 0BH            ;F8 - Control-K
        ZERO15

        DB 0CH            ;F9 - Control-L
        ZERO15

        DB 0EH            ;F10- Control-N
        ZERO15
END
```

FIGURA 2

```
TITLE ACPDAT - ACCEPT DAY/DATE/TIME/ESC KEY/LINE NUM

ENTRY    $ACPDT
EXT      $EVAL,$GETOP,$FLAGS,$ESKEY

DATSIS   EQU  0F248H          ;Endereco da data do sistema
HORSIS   EQU  0000CH          ;Endereco ficticio da hora do sistema
                              ;Consulte o manual de instalacao do MSX-2

         .Z80

$ACPDT:  POP  HL
         INC  HL
         LD   A,(HL)
         INC  HL
         AND  7
         LD   ($FLAGS),A      ;Salva a opcao do ACCEPT
         CALL $EVAL           ;Pega o endereco para onde mover as strings
         LD   A,($FLAGS)      ;Recupera opcao do ACCEPT
         CP   2               ;Seleciona opcao:
         JP   Z,ACDATE        ;1 - ACCEPT xxx FROM DATE (AAMMDD)
         JP   C,ACDAY         ;2 - ACCEPT xxx FROM DAY  (YYJJJ - data juliana)
         CP   4
         JP   C,ACTIME        ;3 - ACCEPT xxx FROM TIME  (HHMMSSCC)
         JP   Z,ACLINE        ;4 - ACCEPT xxx FROM LINE NUMBER (LL)
```

**FIGURA 3**

```
ACESC:   EX   DE,HL
         LD   HL,($ESKEY)     ;$ESKEY contem o valor da tecla
         EX   DE,HL           ;que finalizou o ultimo ACCEPT
ACESC1:  LD   (HL),D          ;Coloca o primeiro digito
         INC  HL
         LD   (HL),E          ;Coloca o segundo digito
         JP   $GETOP          ;Volta

ACLINE:  LD   DE,3030H        ;No Cobol-80 o LINE NUMBER sempre retorna
         JP   ACESC1          ;com o terminal (linha) numero '00'

ACTIME:  PUSH HL              ;Coloca o endereco destino em IX
         POP  IX
         LD   HL,HORSIS       ;Aponta para a hora do sistema
         LD   B,4             ;Prepara para pegar 4 numeros
ACTIM1:  LD   A,(HL)          ;(hora, min, seg, cent)
         PUSH HL              ;Salva o ponteiro
         PUSH BC              ;Salva o contador
         LD   H,0             ;Passa o dado para HL
         LD   L,A
         LD   DE,DEZ          ;Saida em dois digitos
         CALL DECMAL          ;Converte em decimal e guarda no endereco IX
         POP  BC              ;Recupera o contador
         POP  HL              ;Recupera o ponteiro
         INC  HL              ;Incrementa o ponteiro
         DJNZ ACTIM1
         JP   $GETOP          ;Volta
ACDAY:   PUSH HL              ;Coloca endereco destino em IX
         POP  IX
         CALL MOVDAT          ;Move a data do sistema para data interna
         LD   A,(ANO)         ;Pega o ano (armazenado como ANO - 1900)
ACDAY1:  BIT  0,A
         JR   NZ,ACDAY4       ;Ano impar --> nao e' bissexto
ACDAY2:  SRL  A               ;Divide o ano por 4
         SRL  A
         JR   C,ACDAY4        ;Ano nao multiplo de 4 --> nao e' bissexto
ACDAY3:  LD   A,29            ;Ajusta fevereiro para ano bissexto
         LD   (FEV),A
         JR   ACDAY5
ACDAY4:  LD   A,28            ;Inicializa fevereiro
         LD   (FEV),A
ACDAY5:  LD   A,(MES)         ;Pega o mes
         DEC  A               ;Ajusta para mes anterior
         OR   A
         LD   HL,0
         JR   Z,ACDAY7        ;Se o ajuste resultar em zero e' Janeiro
         LD   DE,TABMES       ;Ponteiro para tabela de dias no mes
         LD   B,0             ;Prepara BC para soma (B=0)
ACDAY6:  EX   DE,HL           ;Soma todos os dias ate' o mes anterior
         LD   C,(HL)          ;Prepara BC para soma (C=num dias num mes)
         EX   DE,HL
         ADD  HL,BC
         INC  DE              ;Incrementa ponteiro
         DEC  A               ;Decrementa mes
         JR   NZ,ACDAY6
ACDAY7:  LD   B,0             ;Prepara BC para soma (B=0)
         LD   A,(DIA)         ;Pega o dia
```

```
         LD   C,A             ;Prepara BC para soma (B=0)
         ADD  HL,BC           ;Soma o numero de dias transcorridos do mes
         PUSH HL              ;corrente ao total de dias ate' o mes anterior
         LD   A,(ANO)         ;Pega o ano
         LD   L,A             ;Passa o ano para HL
         LD   H,0
         LD   DE,DEZ          ;Saida em dois digitos
         CALL DECMAL          ;Converte em decimal e guarda no endereco IX
         POP  HL
         LD   DE,CEM          ;Saida de HL em tres digitos
         CALL DECMAL          ;Converte em decimal e guarda no endereco IX
         JP   $GETOP          ;Volta

ACDATE:  PUSH HL              ;Coloca endereco destino em IX
         POP  IX
         CALL MOVDAT          ;Move a data do sistema para data interna
         LD   A,(DE)          ;Usa DE como ponteiro da data interna
         LD   B,3             ;Tres numeros (ano, mes, dia)
ACDAT1:  PUSH BC
         PUSH DE
         LD   DE,DEZ          ;Ponteiro para tabela de valores (DEZ = 2 digitos)
         LD   H,0             ;Prenche HL para conversao em decimal
         LD   L,A
         CALL DECMAL          ;Converte em decimal e guarda no endereco IX
         POP  DE
         INC  DE
         LD   A,(DE)          ;Pega outro numero
         POP  BC
         DJNZ ACDAT1
         JP   $GETOP          ;Volta
MOVDAT:  LD   HL,DATSIS       ;Ponteiro para data do sistema
         LD   DE,DATAUX + 2   ;Ponteiro para data interna
         LD   B,3
MOVDA1:  LD   A,(HL)          ;Pega data do sistema
         INC  HL
         LD   (DE),A          ;Guarda na data interna
         DEC  DE
         DJNZ MOVDA1
MOVDA2:  INC  DE              ;Ajusta Ponteiro
         LD   A,(DE)          ;Pega o ano,
         ADD  A,80            ;Soma 80 ao ano
         CP   100
         JR   C,MOVDA3
         SUB  100             ;Se (ano-1900) > 2 digitos, subtrai 100
MOVDA3:  LD   (DE),A
         RET
```

```
DECMAL:  EX   DE,HL          ;Converte HL em decimal na faixa (0 - 999)
         LD   C,(HL)         ;Prepara BC para subtrair de HL
         LD   B,0
         INC  HL
         EX   DE,HL
         XOR  A              ;Zera contador
DECMA1:  SBC  HL,BC          ;Subtrai
         JR   C,DECMA2       ;Se HL < BC ajusta ou finaliza
         INC  A              ;Contador de iteracoes
         JR   DECMA1
DECMA2:  ADD  A,30H          ;Ajusta contador para digito ASCII
         LD   (IX),A         ;Coloca digito no endereco apontado por IX
         INC  IX             ;Incrementa posicao para o proximo digito
         LD   A,C
         CP   1
         RET  Z              ;Se acabou a conversao, retorna
         ADD  HL,BC          ;senao, ajusta valor
         JR   DECMAL

         DSEG

TABVAL:
         CEN: DB 64H
         DEZ: DB 0AH
         UNI: DB 01H

TABMES:                      ;Tabela do numero de dias nos meses
         JAN: DB 31
         FEV: DB 28
         MAR: DB 31
         ABR: DB 30
         MAI: DB 31
         JUN: DB 30
         JUL: DB 31
         AGO: DB 31
         SET: DB 30
         OUT: DB 31
         NOV: DB 30
         DEC: DB 31          ;ATENCAO: Coloque DEC e nao DEZ!

DATAUX:                      ;Data interna
         ANO: DB 0
         MES: DB 0
         DIA: DB 0
END
```

# Qualidade Internacional

# Made in Brasil.

A qualidade internacional dos disquetes Nashua já é fabricada aqui mesmo no Brasil. Nas três variedades de maior uso na mídia magnética flexível: Disquetes de 5 ¼", 5 ¼" Alta Densidade e 3 ½". Todos com a exclusiva garantia ilimitada Nashua.

**Disquetes Nashua**
O disquete legal.

Fábrica da Nashua no Distrito Industrial da Campo Grande - Rio de Janeiro - Brasil.

Genesis

```
IDENTIFICATION DIVISION.
PROGRAM-ID. TESTE.
AUTHOR.     CARLOS ALBERTO HERSZTERG.

DATA DIVISION.
WORKING-STORAGE SECTION.

***** Caracteres com codigo menor que 32 em Cobol *****
*                                                      *
*    Para efeito de calculo do valor a ser atribuido  *
* ao item declarado binario em dois  bytes  (COMP-0),  *
* faca 256 + cod + 64, pois, fazendo cod=3, temos:     *
*                                                      *
*                    256 = 0100H                       *
*               cod + 64 = 0043H                       *
*               ----------------                       *
*               Carac =    0143H (323 - caracter COPA) *
*                                                      *
*  0143 = CHR$(01)+CHR$(64+3)                          *
*                                                      *
********************************************************

 01 CARACTERES-GRAFICOS COMP-0.
    03 NAIPES.
       05 COPA   PIC 9 VALUE 323.
       05 OURO   PIC 9 VALUE 324.
       05 TREVO  PIC 9 VALUE 325.
       05 ESPADA PIC 9 VALUE 326.

    03 MOLDURA.
       05 TRACO  PIC 9 VALUE 342.
       05 LINHA  PIC 9 VALUE 343.
       05 CANTO1 PIC 9 VALUE 344.
       05 CANTO2 PIC 9 VALUE 345.
       05 CANTO3 PIC 9 VALUE 346.
       05 CANTO4 PIC 9 VALUE 347.

 01 CONSTANTES-GERAIS.
    03 LINHA-IGUAL PIC X(40)  VALUE ALL "=".
    03 LINHA-HIFEN PIC X(40)  VALUE ALL "-".

 01 REGISTRO.
    03 NOME          PIC X(35).
    03 ENDERECO.
       05 RUA       PIC X(20).
       05 NUMERO    PIC 9(05).
       05 APTO      PIC 9(04).
       05 CIDADE    PIC X(15).
       05 UF        PIC X(02).
       05 CEP       PIC 9(05).
    03 TELEFONES.
       05 TELEFONE1.
          10 DDD1   PIC 9(03).
          10 TEL1-1 PIC X(03).
          10 TEL1-2 PIC X(04).
       05 TELEFONE2.
          10 DDD2   PIC 9(03).
          10 TEL2-1 PIC X(03).
          10 TEL2-2 PIC X(04).

 01 VARIAVEIS-GERAIS.
    03 I         PIC 9(03).
    03 CHR       PIC 9 COMP-0.
    03 ALGO      PIC X.
    03 OPCAO     PIC X.
    03 LIMITE    PIC 9(02).

 01 CODIGOS-ESCAPE.
    03 ESC-CODE PIC X(02).
       88 ESC   VALUE "01".
       88 F1    VALUE "02".
       88 F2    VALUE "03".
       88 F3    VALUE "04".
       88 F4    VALUE "05".
       88 F5    VALUE "06".

 01 DADOS-SISTEMA.
    03 DATA-SIS.
       05 ANO      PIC 9(02).
       05 MES      PIC 9(02).
       05 DIA      PIC 9(02).
    03 DATA-JULIANA.
       05 ANO-JUL PIC 9(02).
       05 DIA-JUL PIC 9(03).
    03 HORARIO.
       05 HORA     PIC 9(02).
       05 MIN      PIC 9(02).
       05 SEG      PIC 9(02).
```

FIGURA 4

```
      05 CENT    PTC 9(02).
   03 TERMINAL   PIC 9(02).

01 SCREEN-SECTION-CONSTANTES.
   03 SSC-SETA-CIMA   PIC 9 CONP-0 VALUE 206.
   03 SSC-SETA-BAIXO  PIC 9 COMP-0 VALUE 205.
01 SCREEN-SECTION-CHARS REDEFINES SCREEN-SECTION-CONSTANTES.
   03 SETA-CIMA       PIC X(02).
   03 SETA-BAIXO      PIC X(02).

SCREEN SECTION.
01 CABECALHO.
   03 BLANK SCREEN.
   03 LINE 01 PIC X(40) FROM LINHA-TGUAL.
   03 LINE 03 COLUNN II VALUE "Teste do MSX Cobol".
   03 LINE 05 PIC X(40) FROM LINHA-TGUAL.

01 NENU AUTO.
   03 LINE 10 COLUMN 4 VALUE "[FI] - Tabelas de caracteres".
   03 LINE 12 COLUMN 4 VALUE "[F2] - Caracteres NSX-ASCII".
   03 LINE 14 COLUMN 4 VALUE "[F3] - Edicao em Full Screen".
   03 LINE 16 COLUMN 4 VALUE "[F4] - Dados do sistema".
   03 LINE 18 COLUMN 4 VALUE "[F5] - Finaliza testes".
   03 LINE 22 COLUMN 15 VALUE "Tecle opcao".
   03 LINE 22 COLUMN 27 PIC X TO OPCAO.

01 ENTRADA-DADOS AUTO.
   03 LINE 06 VALUE "<<<< Teste de edicao em Full Screen >>>>".
   03 LINE 08 VALUE "Nome-".
   03         COLUMN 06 PIC X(35) TO NOME.
   03 LINE 10 VALUE "Endereco (Rua, Av)-".
   03         COLUMN 20 PIC X(20) TO RUA.
   03 LINE 12 VALUE "No-".
   03         COLUMN 04 PIC 9(05) TO NUNERO.
   03         COLUMN 11 VALUE "Apto-".
   03         COLUMN 16 PIC 9(04) TO APIO.
   03         COLUMN 22 VALUE "Tels".
   03         COLUMN 27 VALUE "(   )".
   03         COLUMN 28 PIC 9(03) TO DDD1.
   03         COLUMN 33 PTC X(03) TO TELI-I.
   03         COLUMN 36 VALUE "-".
   03         COLUMN 37 PIC X(04) TO TEL1-2.
   03 LINE 13 COLUMN 27 VALUE "(   )".
   03         COLUMN 28 PIC 9(03) TO DDD2.
   03         COLUMN 33 PIC X(03) TO TEL2-1.
   03         COLUNN 36 VALUE "-".
   03         COLUMN 37 PIC X(04) TO TEL2-2.
   03 LINE 15 VALUE "Cidade-".
   03         COLUMN 08 PIC X(15) TO CIDADE.
   03         COLUMN 25 VALUE "UF-".
   03         COLUMN 28 PIC X(02) TO UF.
   03         COLUMN 32 VALUE "CEP-".
   03         COLUMN 36 PIC 9(05) TO CEP.

01 INSTRUCOES.
   03 LINE 16 PIC X(40) FROM LINHA-HTFEN.
   03 LINE 17 COLUMN I2 VALUE "Campo anterior".
   03 LINE 18 COLUMN 2I PIC X(02) FRON SETA-CIMA.
   03 LINE I9 VALUE "Cursor a esquerda <".
   03         COLUMN 23 "> Cursor a direita".
   03 LINE 20 COLUMN 21 PIC X(02) FROM SETA-BAIXO.
   03 LINE 2I COLUMN 12 VALUE "Campo posterior".
   03 LINE 23 VALUE "CTRL-X Apaga campo".
   03         COLUMN 23 VALUE "BS Apaga caracter".
   03 LINE 24 COLUMN 14 VALUE "ESC Finaliza".

01 ESPERA AUTO.
   03 LINE 24 BLANK LINE.
   03         VALUE "Tecle algo para continuar".
   03         COLUMN 27 PIC X TO ALGO.

PROCEDURE DIVISION.
INICIO.
    CALL "COBKEY".
    DISPLAY CABECALHO.
    PERFORM ESCOLHE-OPCAO UNTIL F5.
    CALL "MSXKEY".
FIM.
    STOP RUN.

ESCOLHE-OPCAO.
    DISPLAY (06, 0I) ERASE.
    DISPLAY NENU.
    ACCEPT NENU.
    ACCEPT ESC-CODE FROM ESCAPE KEY.
    IF FI
      PERFORM TABELAS
    ELSE
      IF F2
        PERFORM TESTE32
```

```
      ELSE
        IF F3
          PERFORM ENTRA-DADOS UNTIL ESC
        ELSE
          IF F4
            PERFORM DATA-HORA-SIS.


TABELAS.
    DISPLAY (06, 01) ERASE.
    MOVE 8 TO LIN.
    MOVE 2 TO COL.
    ADD COL 16 GIVING LIMITE.
    DISPLAY (06, 01) "Cods de 32 a 255".
    PERFORM TABELA1 VARYING I FROM 32 BY 1 UNTIL I > 255.
    MOVE 8 TO LIN.
    MOVE 22 TO COL.
    ADD COL 16 GIVING LIMITE.
    DISPLAY (06, 21) "Cods de 288 a 511".
    PERFORM TABELA1 VARYING I FROM 288 BY I UNTIL I > 511.
    DISPLAY ESPERA.
    ACCEPT  ESPERA.
TABELA1.
    MOVE I TO CHR.
    DISPLAY (LIN, COL) CHR.
    ADD I TO COL.
    IF COL = LIMITE
      SUBTRACT 16 FROM LIMITE GIVING COL
      ADD 1 TO LIN.

TESTE32.
    DISPLAY (06, 01) ERASE.
    DISPLAY (06, 01) "<<<< Teste do uso de caracteres MSX >>>>".
    DISPLAY (10, 01) CANTO1.
    MOVE 10 TO LIN.
    PERFORM TESTE32-L VARYING COL FROM 2 BY I UNTIL COL > 39.
    DISPLAY (10, 40) CANTO2.
    DISPLAY (11, 01) TRACO.
    DISPLAY (11, 40) TRACO.
    DISPLAY (12, 01) CANTO3.
    MOVE 12 TO LIN.
    PERFORM TESTE32-L VARYING COL FROM 2 BY I UNTIL COL > 39.
    DISPLAY (12, 40) CANTO4.
    DISPLAY (11, 06) "Caracteres de moldura".
    DISPLAY (14, 16) "NAIPES"
    DISPLAY (15, 14) COPA.
    DISPLAY (15, 17) OURO.
    DISPLAY (15, 20) TREVO.
    DISPLAY (15, 23) ESPADA.
    DISPLAY ESPERA.
    ACCEPT  ESPERA.
TESTE32-L.
    DISPLAY (LIN, COL) LINHA.

ENTRA-DADOS.
    DISPLAY (06, 01) ERASE.
    DISPLAY ENTRADA-DADOS.
    DISPLAY INSTRUCOES.
    ACCEPT ENTRADA-DADOS.
    ACCEPT ESC-CODE FROM ESCAPE KEY.
    DISPLAY (16, 01) ERASE.
    DISPLAY ESPERA.
    ACCEPT ESPERA.

DATA-HORA-SIS.
    DISPLAY (06, 01) ERASE.
    DISPLAY (06, 02) "<<<< Teste dos dados do sistema >>>>".

    ACCEPT DATA-SIS FROM DATE.
    DISPLAY (10, 01) "Data: " DIA "/" MES "/" ANO.

    ACCEPT DATA-JULIANA FROM DAY.
    DISPLAY (12, 01) "Data Juliana: " ANO-JUL "/" DIA-JUL.

    ACCEPT HORARIO FROM TIME.
    DISPLAY (14, 01) "Hora: " HORA ":" MIN ":" SEG "." CENT.

    ACCEPT TERMINAL FROM LINE NUMBER.
    DISPLAY (16, 01) "Terminal no.: " TERMINAL.

    DISPLAY ESPERA.
    ACCEPT  ESPERA.
```

```
1 - Arquivo batch CL.BAT
Compila e executa o link sem geração do arquivo de Debug

        COBOL %1,%2=%1/D
        Pause, Control-C para abandonar ou...
        L80 %1,%1/N/E
        Pause, Pronto para iniciar execucao
        %1

Uso: CL nomeprog [nomelist]
     onde "nomeprog" é o nome do arquivo-fonte e "nomelist"
     (opcional) é o nome do arquivo-listagem, que pode ser TTY:,
     LST: ou qualquer nome válido para um arquivo.

Obs: Serão criados os seguintes arquivos:
     - nomeprog.REL
     - nomeprog.COM
     - nomelist.PRN (se declarado arquivo em disco)


2 - Arquivo batch CLD.BAT
Compila e executa o link com geração do arquivo de Debug

        COBOL %1,%2=%1
        Pause, Control-C para abandonar ou...
        L80 %1,%1/N/E
        Pause, Pronto para iniciar execucao
        %1

Uso: CLD nomeprog [nomelist]

Obs: Serão criados os seguintes arquivos:
     - nomeprog.REL
     - nomeprog.COM
     - nomelist.PRN (se declarado arquivo em disco)
     - progr-id.DBG (progr-id é o nome declarado no PROGRAM-ID)


3 - Arquivo batch CLL.BAT
Compila e executa o link com geração do arquivo de Debug e criação  automática  do
arquivo-listagem em disco, simultaneamente à listagem em tela.

        COBOL %1,TTY:%1.N
        Pause, COntrol-C para abandonar ou...
        L80 %1,%1/N/E
        Pause, Pronto para iniciar execução
        %1

Uso: CLL nomeprog

Obs: Mesmos arquivos gerados pelo CLD.BAT.
```

FIGURA 5

# MSX LORDS

SILVIO CHAN

Este jogo foi feito para dois jogadores, sendo que o jogador 1 deverá optar por usar as setas de cursor ou o joystick 1 e o jogador 2 deverá utilizar o joystick 2.

O jogador 1 é representado por uma barreira na parte superior da tela, enquanto o jogador 2 é representado por outra barreira na parte inferior da tela.

O jogo é do tipo paredão, no qual cada jogador deverá defender as pequenas coroas incrustadas em seus domínios de uma esfera móvel.

Cada jogador poderá realizar os seguintes movimentos: esquerda, direita, para cima e para baixo.

Os movimentos para cima e baixo só podem ser feitos na pista central da tela.

Toda vez que um jogador bloquear a esfera, ele irá perder 10 unidades de energia (no início do jogo temos 200 unidades de energia), que poderão ser recuperadas apanhando as cápsulas energéticas que, de vez em quando, aparecem no centro da pista central. Cada cápsula fornece 20 unidades de energia

O vencedor será aquele que impedir a destruição de qualquer uma das coroas de seus domínios e não atingir o nível zero de unidades de energia.

### DIGITAÇÃO

Deve-se tomar cuidado na digitação das linhas DATA que contêm os códigos da rotina em Linguagem de Máquina. Recomendo que gravem o programa antes de testá-lo.

```
10 ' ==========================
20 '
30 '       * MSX LORDS *
40 '
50 '  (C) 1990 BY SCHAN SOFT
60 '
70 ' ==========================
80 '
90 ' INICIALIZAÇAO
100 '
110 KEYOFF:COLOR 15,1,1:SCREEN1,3,0:WIDT
H32
120
130 'ROTINA EM L.M.
140 '
150 DATA 3A,F8,F7,FE,1,28,4C,FF,2,28,5E,
FE,3,28,68,FE,4,28,78,FE,5,28,47,FE,6,28
,55,FE,7,28,62,FE
160 DATA 8,28,6F,FE,9,CA,99,D0,FE,A,CA,B
1,D0,FE,B,CA,CA,D0,FE,C,CA,E3,D0,FE,D,CA
,2B,D1,21,0,8,2B
170 DATAE5,CD,4A,0,57,F,B2,CD,4D,0,54,7D
,E1,B2,FE,0,20,ED,C9,CD,FB,D0,11,20,0,ED
,52,C3,13,D1,CD,7
180 DATA D1,11,20,0,ED,52,C3,1F,D1,CD,FB
,D0,23,C3,13,D1,CD,7,D1,23,C3,1F,D1,CD,F
B,D0,11,20,0,19,C3,13
190 DATA D1,CD,7,D1,11,20,0,19,C3,1F,D1,
CD,FB,D0,2B,C3,13,D1,CD,7,D1,2B,C3,1F,D1
,3E,0,32,4,D3,2A,0
200 DATA D3,11,21,0,19,CD,4A,0,FE,D8,C0,
3E,FF,32,4,D3,C9,3E,0,32,5,D3,2A,2,D3,11
,1F,0,ED,52,CD,4A
210 DATA 0,FE,D8,C0,3E,FF,32,5,D3,C9,3E,
0,32,4,D3,2A,0,D3,11,1F,0,ED,52,CD,4A,0,
FE,D8,C0,3E,FF,32
220 DATA 4,D3,C9,3E,0,32,5,D3,2A,2,D3,11
,21,0,19,CD,4A,0,FE,D8,C0,3E,FF,32,5,D3,
CD,2A,0,D3,1,4
230 DATA 0,3E,20,CD,56,0,C9,2A,2,D3,1,4,
0,3E,20,CD,56,0,C9,1,4,0,3E,C8,CD,56,0,2
2,0,D3,C9,1
240 DATA 4,0,3E,D0,CD,56,0,22,2,D3,C9,3A
,8,D3,FE,0,CA,5B,D1,FE,1,CA,79,D1,FE,2,C
A,97,D1,2A,6,D3
250 DATA CD,B5,D1,CA,FF,D1,2A,6,D3,CD,CA
,D1,CA,37,D2,2A,6,D3,CD,4A,D2,CA,5A,D2,C
3,60,D2,2A,6,D3,CD,BF
260 DATA D1,CA,7,D2,2A,6,D3,CD,D4,D1,CA,
1F,D2,2A,6,D3,CD,3F,D2,CA,54,D2,C3,60,D2
,2A,6,D3,CD,B5,D1,CA
270 DATA F,D2,2A,6,D3,CD,CA,D1,CA,27,D2,
2A,6,D3,CD,4A,D2,CA,5A,D2,C3,60,D2,2A,6,
D3,CD,BF,D1,CA,17,D2
280 DATA 2A,6,D3,CD,D4,D1,CA,2F,D2,2A,6,
D3,CD,3F,D2,CA,54,D2,C3,60,D2,11,20,0,19
,CD,4A,0,FE,C0,C9,11
290 DATA 20,0,ED,52,CD,4A,0,FE,B8,C9,11,
20,0,19,CD,4A,0,FE,D0,C9,11,20,0,ED,52,C
D,4A,0,FE,C8,C9,3E
300 DATA 0,32,8,D3,C3,60,D2,3E,1,32,8,D3
,C3,60,D2,3E,2,32,8,D3,C3,60,D2,3E,3,32,
8,D3,C3,60,D2,3E
310 DATA 20,CD,4D,0,C3,EF,D1,3E,20,CD,4D
,0,C3,E7,D1,3E,20,CD,4D,0,C3,DF,D1,3E,20
,CD,4D,0,C3,F7,D1,3E
320 DATA 1,32,9,D3,C3,E7,D1,3E,2,32,9,D3
,C3,DF,D1,3E,1,32,9,D3,C3,F7,D1,3E,2,32,
9,D3,C3,EF,D1,11
330 DATA 20,0,ED,52,CD,4A,0,FF,B9,C9,11,
20,0,19,CD,4A,0,FE,C1,C9,3E,FF,32,A,D3,C
9,3E,FF,32,8,D3,C9
340 DATA 2A,6,D3,3E,20,CD,4D,0,3A,8,D3,F
E,0,CA,80,D2,FE,1,CA,87,D2,FE,2,CA,8D,D2
,11,21,0,19,18,14
350 DATA 11,21,0,ED,52,18,D,11,1F,0,19,1
8,7,11,1F,0,ED,52,18,0,22,6,D3,3E,C9,CD,
4D,0,C9
360 '
370 ' BLOCOS GRAFICOS
380 '
390 DATA 127,65,95,95,95,95,127,0
400 DATA 66,66,102,102,126,102,126,0
410 DATA 255,255,255,0,0,255,255,255
420 DATA 0,60,126,126,126,126,60,0
430 DATA 126,195,165,153,153,165,195,126
440 '
450 ' INSTALA ROTINA EM L.M.
460 '
470 FOR I=&HD000 TO &HD29C : READ A$ : P
OKE I,VAL ("&H"+A$):NEXT
480 '
490 ' DEFINE VARIAVEIS
500 '
510 CLEAR:DEFINTA-Z:R=RND(-TIME)
520 E1=200:E2=E1:X1=14:X2=X1:Y1=4:Y2=16
530 POKE&HD306,&HA5:POKE&HD307,&H18:POKE
&HD308,3:POKE&HD309,0:POKE&HD30A,0:POKE&
HD30B,0
540 I=&HD300:POKEI,&HBE:POKEI+1,&H18:POK
EI+2,&HE:POKEI+3,&H1A
550 '
560 ' TELA DE ABERTURA
570 '
580 SCREEN2:OPEN"GRP:"AS#1
590 FOR I=0 TO 1:DRAW"BM"+STR$(20+I)+",2
0":PRINT#1,"Defenda sua fortaleza em...
600 DRAW"BM"+STR$(8+I)+",180":PRINT#1,"C
opyright 1990 by Silvio Chan.":NEXT
610 FOR I=0TO2:DRAW"BM"+STR$(66+I)+",65C
2D60R20BR10BU60G5D50F5R10E5U50H5L10BR20B
D60U60R15F5D20G5L15F20D10BR5U60R15F5D50G
5L15BR25RR15E5U20H5L10H5U20E5R15":NEXT
620 IF INKEY$<>""THEN 620
630 A$=INPUT$(1)
640 '
650 ' REDEFINE BLOCOS
660 '
670 SCREEN1:DEFUSR=&HD000:Z=USR(0)
680 RESTORE 390
690 FOR I=1472 TO 1487:READA:VPOKEI,A:NE
XT
700 RESTORE 390
710 FOR I=1536TO1551:READA:VPOKEI,A:NEXT
720 FORI=1664TO 1671:READA:VPOKEI,A:NEXT
730 RESTORE 410
740 FOR I=1600TO1615:READA:VPOKEI,A:NEXT
750 FORI=1728TO1735:READA:VPOKEI,A:NEXT
760 SPRITE$(0)=STRING$(16,128)+STRING$(1
6,1)
770 VPOKE8215,113:VPOKE8216,33:VPOKE8217
,209:VPOKE8218,161:VPOKE8219,145
780 '
790 ' TELA DO JOGO
800 '
810 FOR I=0 TO 95: PRINT CHR$(184);:NEXT
820 FOR I=0 TO 2 : LOCATE 5+I,0 : PRINT
CHR$(185);
830 LOCATE 13+I:PRINT CHR$(185);
840 LOCATE 16+I:PRINT CHR$(185);
850 LOCATE 24+I:PRINT CHR$(185);:NEXT
860 LOCATE 0,18:FOR I=0 TO 95:PRINT CHR$
(192);:NEXT
870 FOR I=0 TO 2: LOCATE 5+I,20:PRINT CH
R$(193);
880 LOCATE 13+I:PRINT CHR$(193);
890 LOCATE 16+I:PRINT CHR$(193);
900 LOCATE 24 +I: PRINT CHR$(193);: NEXT
910 FOR I=6816 TO 6847:VPOKE I,23:NEXT
920 PUTSPRITE0,(112,41),14,0
930 PUTSPRITE1,(112,61),14,0
940 PUTSPRITE2,(112,93),14,0
950 GOSUB 1290
960 '
970 ' PROCESSAMENTO CENTRAL
980 '
990 LOCATE 5,5: PRINT CHR$(201)
1000 LOCATE X1,Y1:PRINT CHR$(200)CHR$(20
0)CHR$(200)CHR$(200)
1010 LOCATE X2,Y2:PRINT CHR$(208)CHR$(20
8)CHR$(208)CHR$(208)
1020 A=STICK(0)OR STICK(1)
1030 B=STICK(2)
1040 IFA=1ANDY1>4THENZ=USR(11):Z=USR(1):
Y1=Y1-1
```

```
1050 IF B=1ANDY2=14ANDY2>10 AND Y2>Y1+2
THEN Z=USR(10):Z=USR(5):Y2=Y2-1
1060 IFA=3 AND Y1=4 AND X1 <28 THEN Z=US
R(2):X1=X1+1
1070 IF B=3 AND Y2=16 AND X2<28 THEN Z=U
SR(6):X2=X2+1
1080 IF A=5 AND X1=14 AND Y1<10 AND Y1<Y
2-2 THEN Z=USR(9): Z=USR(3): Y1=Y1+1
1090 IF B=5 AND Y2<16 THEN Z=USR(12): Z=
USR(7):Y2=Y2+1
1100 IF A=7 AND Y1=4 AND X1>0 THEN Z=USR
(4):X1=X1-1
1110 IF B=7 AND Y2=16 AND X2>0 THEN Z=US
R(8): X2=X2-1
1120 IF PEEK(&HD304)=255 THEN E1=E1+20:P
OKE &HD304,0
1130 IF PEEK (&HD305)=255 THEN E2=E2+20:
 POKE &HD305,0
1140 IF PEEK (&HD309)=1 THEN E1=E1-10 :
POKE &HD309,0
1150 IF PEEK (&HD309)=2 THEN E2=E2-10 :
POKE &HD309,0
1160 IF E1=0 THEN 1490
1170 IF E2=0 THEN 1500
1180 IF PEEK (&HD30A)=255 THEN 1490
1190 IF PEEK (&HD30B)=255 THEN 1500
1200 R=RND(1)*100:IF R<3 AND Y1<>10 AND
Y2 <> 10 THEN LOCATE 15,10 : PRINT CHR$(
216)
1210 LO=(PEEK(&HD307)*256+PEEK(&HD306))-
6144
1220 XB=LO AND 31 : YB=(LO-XB)/32
1230 IF XB=0 OR XB=31 OR YB=0 OR YB=20 T
HEN GOSUB 1340
1240 Z=USR(13)
1250 GOSUB 1290: GOTO 1020
1260 '
1270 ' PLACAR
1280 '
1290 LOCATE 0,22:PRINT"PLAYER1"E1:PRINT"
PLAYER2"E2;
1300 RETURN
1310 '
1320 'LIMITES DA TELA
1330 '
1340 IF XB=0 AND YB=0 THEN POKE &HD308,3
 : RETURN
1350 IF XB=0 AND YB=20 THEN POKE &HD308,
2 : RETURN
1360 IF XB=31 AND YB=0 THEN POKE &HD308,
1 : RETURN
1370 IF XB=31 AND YB=20 THEN POKE &HD308
,0 : RETURN
1380 IF XB=0 AND PEEK (&HD308)=0 THEN PO
KE &HD308,2:RETURN
1390 IF XB=0 AND PEEK (&HD308)=1 THEN PO
KE &HD308,3 : RETURN
1400 IF XB=31 AND PEEK (&HD308)=2 THEN P
OKE &HD308,0 : RETURN
1410 IF XB=31 AND PEEK (&HD308)=3 THEN P
OKE &HD308,1 : RETURN
1420 IF YB=0 AND PEEK (&HD308)=0 THEN PO
KE &HD308,1 : RETURN
1430 IF YB=0 AND PEEK (&HD308)=2 THEN PO
KE &HD308,3 : RETURN
1440 IF YB=20 AND PEEK(&HD308)=1 THEN PO
KE &HD308,0 : RETURN
1450 POKE &HD308,2: RETURN
1460 '
1470 ' FIM DE JOGO
1480 '
1490 GOSUB 1290 : LOCATE 0,10 : PRINT "P
LAYER 1 DESTROYED...":FORI=0 TO 5000 : N
EXT : GOTO 510
1500 GOSUB 1290:LOCATE 0,10 : PRINT"PLAY
ER 2 DESTROYED...":FORI=0 TO 5000 : NEXT
 : GOTO 510
1510 '
1520 ' VARIAVEIS USADAS
1530 '
1540 X1,X2 - X DAS BARREIRAS
1550 Y1,Y2 - Y DAS BARREIRAS
1560 XB,YB - COORD. DA BOLA
1570 E1,E2 - ENERGIA
1580 A,B - JOYSTICKS
```

PROJETO

# MSXDEBUG

## PARTE XI

Sérgio Duric Calheiros

Como último artigo dedicado ao processo de conversão de programas, neste mês discutiremos como aproveitar as telas que acompanham os programas que estão sendo convertidos. Não são raros os programas que possuem telas gráficas ou procedimentos semelhantes durante a carga. Naturalmente, não poderiamos encher os olhos dos leitores com mil maneiras de economizar espaço em disco, tendo que pagar um alto preço para consegui-la.

O processo de conversão de telas, conjuntamente com seus programas, é bem semelhante ao processo de conversão dos programas sem tela. A única dificuldade é reconhecer as partes úteis de uma tela.

De uma maneira análoga, numa conversão casual, os procedimentos a serem tomados são exatamente uma imitação dos passos tomados pelo BASIC. No momento em que uma tela for encontrada, é preciso executar a rotina que faz a transferência da memória para o vídeo e, a seguir, carregar os demais blocos como descrito anteriormente.

Para adaptar a tela ao novo ambiente, basta utilizar o processo de carregamento do DOS. Geralmente, nos programas ainda não convertidos, a tela é o primeiro bloco a ser carregado e executado. Nos programas já convertidos, a tela sempre será a primeira a ser executada. Logo após o carregamento da tela, deve-se carregar o bloco relativo ao programa propriamente dito. Esse processo dá um aspecto extremamente profissional ao micro, pois livra a imagem de dependência do BASIC.

Mas como fazer para que a execução do programa seja feita após o carregamento da tela? Como fazer para executar os blocos separadamente?

O fato é que devemos tratar cada parte, programa e tela, como entidades diferentes, sendo executadas em horas diferentes. O programa principal deve estar num bloco único, convertido, usando-se uma das maneiras explicadas anteriormente. A tela deve ser outro bloco executável, mas que tenha a habilidade de carregar o arquivo sobressalente, ou o programa propriamente dito, automaticamente após sua execução.

Essa manobra pode ser feita através de um RUNTIME adicionado às rotinas de transferência da tela que, além de fazer tudo que o RUNTIME adicionado pelo SAVECOM faz, ainda carrega o arquivo sobressalente.

Com isso, concluímos que a diferença entre um arquivo que ao ser executado apresenta uma tela e carrega outro arquivo e outro que simplesmente executa, está no RUNTIME.

Dessa maneira, para converter uma tela, deve-se usar o comando SAVESCR que colocará o RUNTIME adequado, ao invés do comando SAVECOM, que coloca o RUNTIME que tem apenas as rotinas de transferência e de execução.

```
BLOCO 1

590D CD A5 08 CD FA 08 22 05
5915 19 11 AF 18 CD 18 0B CD
591D 24 0B CD 62 0A 21 00 00
5925 E5 CD 27 09 28 04 CD FA
592D 08 E3 E1 22 D4 18 7C B5
5935 3E 01 28 02 3E 31 32 D3
593D 18 11 B5 18 CD 18 0B CD
5945 24 0B CD 62 0A 21 00 00
594D E5 CD 27 09 28 04 CD FA
5955 08 E3 C1 2A 89 0D 09 22
595D E0 18 2A 8B 0D 09 22 DB
5965 18 11 BC 18 CD 18 0B CD
596D 24 0B CD 62 0A CD 9A 08
5975 CD FA 08 7D 32 01 19 3E
597D 21 32 D6 18 21 48 01 22
5985 D7 18 21 C0 01 22 E7 18
598D 21 01 00 22 C7 18 21 17
5995 0E 11 59 1A 01 08 00 ED
599D B0 21 E9 19 01 03 00 ED
59A5 B0 CD 9F 18 CD 66 07 CD
59AD 72 07 CD 5D 06 21 C5 18
59B5 11 80 00 01 48 00 ED B0
59BD 21 EC 19 01 38 00 ED B0
59C5 E5 11 5C 00 CD 2B 07 E1
59CD 11 80 00 01 40 00 ED B0
59D5 2A 89 0D 01 40 00 ED B0
59DD 22 89 0D 11 5C 00 CD 2B
59E5 07 C3 89 06 4F 56 52 21
59ED 5C 00 E5 54 5D 13 01 24
59F5 00 36 00 ED B0 D1 13 21
59FD B5 01 01 0B 00 ED B0 11
5A05 5C 00 0E 0F CD 05 00 B7
5A0D C2 00 00 21 7A 01 11 80
5A15 00 01 80 00 ED B0 C3 80
5A1D 00 11 00 01 D5 0E 1A CD
5A25 05 00 11 5C 00 0E 14 CD
5A2D 05 00 FE 02 CA 00 00 B7
5A35 D1 20 07 21 80 00 19 EB
5A3D 18 E2 11 00 08 06 00 10
5A45 FE 1B 7A B3 20 F9 21 00
5A4D 01 E5 DB A8 E6 FF 32 F4
5A55 FA C3 04 FB 00 00 00 00

Soma total:006EB8
```

Analogamente ao comando SAVECOM, o comando SAVESCR cria, em disco, o programa (no caso a tela) que está numa área de memória, com todas as informações como os endereços, posição da pilha (STACK), deslocamento relativo (OFFSET) e ambiente. Por outro lado, o arquivo a ser carregado deverá ter o mesmo nome do arquivo que contém a tela, mas com extensão OVR. Esse arquivo, se encontrado, será carregado e executado no endereço 100H como faz o próprio DOS.

Para simplificar a descrição do processo, usaremos um pequeno exemplo. Suponhamos que o programa que será convertido possua, além da tela, apenas um bloco. Reconhecendo qual é o bloco principal, basta convertê-lo da forma usual, usando o comando SAVECOM. O arquivo criado no disco terá a extensão COM, podendo ser executado a qualquer momento. Para que seja carregado pela tela, deve ser renomeado para que tenha extensão OVR. Aliás, este é um ponto importante, pois o comando SAVESCR também cria um arquivo com extensão COM, podendo sobrepor o antigo, já que, teoricamente, devem possuir o mesmo nome. A sintaxe do comando SAVESCR é idêntica à sintaxe do comando SAVECOM. Apenas fique atento aos seus procedimentos, evitando qualquer confusão com um ou outro comando.

Após renomear o programa, através do DOS, retorne ao MSXDEBUG, e converta a tela como SAVESCR. Todas as informações que serão passadas ao comando SAVESCR, serão relativas à tela. O usuário deve se preocupar com o bloco principal somente na hora de convertê-lo ao usar comando SAVECOM.

A implementação do comando, não é mais um mistério, dispensando maiores detalhes. Apenas fiquem atentos aos endereços do bloco e à definição da chamada.

Pode ser que, para alguns, a inclusão de uma tela em um programa seja algo um tanto supérfluo, servindo apenas para enfeitar e ocupar espaço do disco. Mas convenhamos que, diante do espaço economizado através das conversões, é rasoável manter uma coleção de programas aceitável. Vale ressaltar ainda, que uma tela convertida também economiza espaço. Até a próxima.

# O MUNDO É DOS EXPERTS.

Quer dizer que você quer ser o melhor? Quer entrar no futuro pela porta da frente? Chegaram os novos Experts da Gradiente, os primeiros micros para quem entende e para quem não entende de computador. Para quem entende, o Expert DD Plus é o único no Brasil com disk-drive de 3,5 polegadas embutido. Para quem não entende, o Expert Plus é muito fácil de operar e até vem com um programa na memória que mostra um pouco de tudo o que você pode fazer com ele. Isso significa que é só comprar e começar a usar. Os Experts têm disponíveis todos os periféricos necessários para o seu sistema não parar de crescer, como o Multi-Modem, os Cartões 80 Colunas e muitos outros. Sem falar nos 1500 programas para você fazer desde gráficos até música, do controle financeiro de uma pequena empresa até jogos da loto, aprender francês, controlar o saldo do banco ou escrever um livro. Com um Expert você tem acesso ao Videotexto, a outros Experts via linha telefônica e a um processador de textos que vai deixar você sem palavras. Ele é agenda, arquivo, central de informações, video game e até terminal de PC. Para conhecer melhor os novos micros da Gradiente, peça uma demonstração. Afinal, hoje você é expert ou não é nada.

Produzido na Zona Franca de Manaus

SPECTRO

ANALISADOR DE AUDIO

PROJETO SPECTRO

Por: Sergio Duric Calheiros

# O PSG SOB UM NOVO PONTO DE VISTA

Desde o lançamento dos MSX nacionais, inúmeros artigos sobre o PSG foram publicados. Todos tentando dar um enfoque diferente sobre o assunto, mas que invariavelmente falavam sempre a mesma coisa. Ensinavam como fazer o PSG gerar som. Alguns se limitavam apenas a repetir tudo aquilo que já estava explicado no próprio manual do micro.

Ao contrário do que muita gente pensa, o PSG não serve apenas para gerar sons ou ruídos. E ele é responsável por outras atividades do computador, como o tratamento dos sinais enviados por um dos dois joysticks conectados a ele ou por um dos 12 paddles que ele pode controlar simultaneamente. Evidentemente é um exagero, mas o PSG pode fazer isso.

Além disso, o PSG também gerencia o set de caracteres do teclado em máquinas japonesas. Nos nacionais, esta função não é necessária e, portanto, não existe.

Uma das funções mais importantes atribuídas ao PSG é o processamento dos sinais enviados pelo cassete. Não é à toa que o sistema de gravação e leitura em fita cassete, dos micros da linha, é um dos melhores e mais confiáveis que existem. Neste sistema, o reconhecimento da frequência do sinal do cassete é feito por software, mas o responsável pelo tratamento do sinal, que é entregue ao Z-80, é o PSG. Este padrão, adotado pelos micros MSX, é conhecido como FSK ou Frequency Shift Keying.

### Linguagem de máquina e Eletrônica.

Quando se fala em linguagem de máquina, não se pode deixar de falar em eletrônica. Afinal, o que é linguagem de máquina senão códigos binários formados por sinais elétricos? É claro que estamos falando de níveis baixíssimos de informação. Na prática é muito comum usarmos mnemônicos, números prontos e no mínimo um assemblador. Entretanto, dependendo da situação, não podemos esquecer mais esse aspecto dessa linguagem.

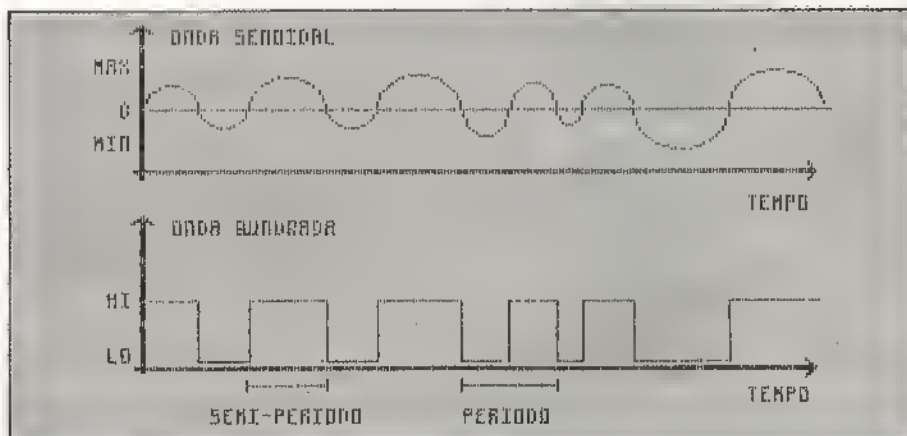
No microcomputador, não basta apenas existir um processador e memória. Muito mais é necessário para que tudo aconteça como acontece. Os circuitos periféricos devem estar em perfeita sintonia com o resto do sistema. Os sinais que passam através das linhas de comunicação devem ser entregues de forma clara e precisa, para que possam ser bem aproveitados por quem os recebe.

É exatamente neste ponto que devemos nos concentrar se quisermos aproveitar mais essa característica do computador. Ou, mais especificamente, do PSG.

O sinal presente na porta do cassete, se resume numa sequência de níveis elétricos altos e baixos. Após ser digitalizado, o sinal se torna disponível à saída de uma das portas do PSG para que seja lido. Entretanto, o tempo em que este sinal fica disponível é curtíssimo. Somente uma linguagem rápida o suficiente, como é a linguagem de máquina, é capaz de ler, processar a informação e voltar a tempo de esperar a próxima.

### O projeto SPECTRO

Tudo que falamos até aqui é apenas uma introdução a um novo projeto que iniciamos a partir deste número. Queremos dar ao usuário dessas máquinas, mais uma razão que o diferencie dos usuários comuns de equipamentos de outras linhas.

O Spectro é um programa que lê e compara a frequência dos sinais presentes na porta do cassete, sejam eles analógicos ou digitais. Com esse programa, poderemos ajustar o azimute do cassete e até analisar o espectro sonoro de um sinal de áudio. O espectro sonoro é a curva instantânea da gama de frequências que o sinal apresenta. Podemos representar isso graficamente por uma série de barras que indiquem se determinada frequência está ou não presente.

DIGITALIZAÇÃO DA ONDA

Para que não haja dúvidas de que o programa é legítimo, ou seja, desenvolvido por mim, procuraremos apresentar todas as informações possíveis a respeito dele.

Como os demais projetos dessa linha, o Spectro será apresentado por partes e em cada parte uma novidade.

**Velocidade, frequência e tempo**

O papel do PSG é nos entregar o sinal da forma que nos interessa, ou seja, digitalizado. Isso está ilustrado na figura 1. Notem que o período da onda, nesse caso semi-período, não é alterado. O sinal digital imita seu analógico equivalente. E é exatamente isso que precisamos para analisar a frequência.

Se observarmos a figura novamente, veremos que uma das informações foi perdida. As diversas amplitudes que um sinal analógico pode ter, foram transformadas em apenas uma. Um sinal digital tem apenas uma amplitude, ou seja, apenas um valor para o nível alto além do nível zero para o nível baixo. Não que isso seja um empecilho, mas apenas não poderemos nos basear nessa informação ao construir o gráfico de barras, simplesmente por sua inexistência.

A frequência é uma grandeza que está diretamente ligada ao tempo. Por isso, para termos alguma coerência naquilo que fazemos com o dado que recebemos, devemos assumir alguma base de tempo como válida . Felizmente, o computador já possui uma. O próprio clock do processador é uma excelente base de tempo, constante e precisa.

Examine novamente a figura 1. Observe que a onda é um constante sobe-e-desce, que também é chamado de batimento, ou comprimento de onda. A frequência da onda é, então, o número de vezes que a onda faz este batimento a cada segundo. O período, como podemos deduzir, é então o tempo que cada batimento leva para concluir seu ciclo. Consequentemente, o período é o inverso da frequência.

De um modo geral, as frequências podem assumir uma faixa enorme de valores. Por exemplo, a rotação da terra em torno do sol tem um período de 1 ano. A frequência, neste caso, é baixíssima. A luz, sendo uma onda eletromagnética, também tem uma frequência. Esta frequência é da ordem de 10 elevado a 14 Hertz, ou seja, um número seguido de 14 zeros.

Hertz é a unidade em que medimos a frequência.

Felizmente, a faixa de frequência com que vamos trabalhar, varia de algumas unidades até uns 20 kHz. Esta é a faixa que o ser humano pode ouvir. Digo felizmente, porque se essa faixa fosse um pouco maior, nem a linguagem assembler seria capaz de acompanhar essa velocidade. Pelo menos a do MSX.

Outro conceito que devo introduzir, é a velocidade de processamento em tempo real. Isso quer dizer que não pode haver atraso durante o processamento, sob o risco de invalidarmos a operação que está sendo executada. Um exemplo é quando o micro acessa o drive. Todas as interrupções são desabilitadas e não há leitura do teclado nesse momento. Nosso caso é idêntico. Isso trouxe um problema que foi solucionado justamente com a própria causa, o que será devidamente esclarecido nos próximos artigos.

**O sinal**

O sinal que entra pela porta do cassete, muda constantemente. Mas, sempre que quisermos analisá-lo, ele estará lá, seja qual for seu valor. Ele não espera que comecemos a lê-lo.

Normalmente, a execução de um programa é feita de maneira sequencial. Além da rotina que lê o sinal, existem outras como as que atualizam o VDP e as rotinas auxiliares. Desse modo, sabemos que não é o tempo todo que o sinal está sendo lido. Além do mais, quando começamos a fazê-lo exatamente no início do período da onda.

Para que tenhamos um mínimo de precisão, temos que garantir que estamos considerando o sinal somente a partir do início do ciclo. Para isso, basta começar a ler o sinal — esteja ele em qualquer parte do período — e assim que transicionar, começar a contar.

Como já disse, o próprio tempo que o Z-80 leva para executar uma instrução é uma maneira de contar eventos. Então, para saber o perído em que o ciclo da onda permaneceu na porta do PSG, devemos verificar se ele ainda está lá repetidas vezes até que ele mude.

Assim que detectarmos essa mudança, teremos nas mãos um valor que corresponde à frequência lida. E assim, de frequência em frequência, o programa constrói o gráfico. Pegou?

**Os valores**

Falando de valores reais, tome como exemplo uma frequência de apenas 32 Hz. Fazendo as contas, veremos que o micro perde cerca de 31 mili segundos para reconhecê-la, o que é uma eternidade em termos da linguagem de máquina. Já para uma frequência maior como a de 16.000 Hz, o micro leva apenas 62 micro segundos. Na prática, isso equivale à execução de apenas uma dezena de instruções.

Logo, estamos lidando com uma faca de dois gumes. De um lado, com uma frequência baixa há um desperdício de tempo, deixando o programa lento. Por outro, com frequências muito altas, quase não há tempo do sinal ser lido.

O fato de trabalharmos com um micro com uma velocidade razoável, não muda muita coisa. Mesmo que o micro fosse 20 vezes mais rápido, o período de frequências baixas não iria mudar. Isso é o que chamo de tempo real.

O processamento do sinal lido é feito de forma bem simples. Se você já teve a oportunidade de observar um equalizador de áudio, viu que a frequência mínima que ele manipula é 32 Hz e a máxima é 16K Hz. Além disso, o salto de uma faixa para outra é em progressão geométrica de razão 2, ou seja, 32, 64, 125, 250, etc.

Com o valor do sinal em mãos, basta compará-lo com uma tabela, para depois jogá-lo na faixa que ele pertence.

Esta tabela não tem valores aleatórios. Seus valores foram calibrados com uma parafernália que reuniu dois computadores MSX, um mixer, um amplificador e um equalizador, um dos computadores foi usado para gerar um sinal puro, com apenas uma harmônica, usando um dos canais do PSG. O outro foi usado para analisar o sinal a construir a tabela. As medições foram feitas para todas as frequências separadamente. O equalizador ajudou a filtrar qualquer outra frequência ou ruído indesejável. Nada disso seria possível sem o apoio e a assistência do meu amigo José Raul de Moraes, que esteve presente até a última versão do sistema.

No próximo número, veremos como funciona a rotina principal, responsável pelo reconhecimento da frequência. Com ela, introduziremos as bases do programa Spectro. Espero que tudo que foi dito aqui tenha servido para aguçar a curiosidade de cada um e para mostrar que não há limites para quem leva algo a sério.

Luis Carlos Barbosa de Oliveira é técnico em Hardware e Software e, atualmente, atua na área de manutenção e desenvolvimento de programas comerciais para as linhas MSX e PC, na empresa Cheyenne Advanced Systems.
Para a linha MSX, desenvolveu o programa Aquarela, um poderoso editor gráfico.

**CPU - Qual foi seu primeiro contato com a Informática?**
Luis Carlos - O primeiro contato se deu em 1982, quando ainda era office-boy e me impressionei com os recursos de um micro TRS-80.

**CPU - E o seu primeiro contato com um micro?**
Luis Carlos - Foi com um TK-85 e, logo em seguida, com um TK 90X.

**CPU - Quando você sentiu que era o momento de passar para o MSX?**
Luis Carlos - Quando um amigo, que já possuia um MSX na época do lançamento, me mostrou todos os recursos disponiveis, tanto da máquina como da diversificação dos periféricos que as duas empresas, na época, prometiam aos usuários.

**CPU - O que o levou a ser um programador?**
Luis Carlos - Na época, o nivel de literatura e apoio ao MSX era altamente deficiente e achei um absurdo que uma máquina com tantos recursos fosse condenada a mais um videogame de teclado.

**CPU - Como videogame de teclado?**
Luis Carlos - Simples. Em 1985, todos os anúncios de softwares se voltavam exclusivamente para o lazer. Por isso, achei injusto que um micro semi-profissional fosse tratado como um brinquedo, sendo o mesmo um filho caçula de 8 bits do PC.

**CPU - Então podemos considerar que o MSX é um micro profissional?**
Luis Carlos - Por que não? Afinal de contas, hoje, contamos com softwares, de nivel nacional e internacional, tais como Pagemaker, Cadastros Profissionais, dBase II plus, Super Calc II, Linguagens (C, Turbo Pascal, Cobol, etc), Wordstar III e o Aquarela, que é de minha autoria.

**CPU - Você acha que uma empresa pode trabalhar com um MSX?**
Luis Carlos - Qualquer empresa de pequeno ou médio porte pode trabalhar com um MSX. Tenho um amigo na cidade de Suzano, estado de São Paulo, dono de uma revendedora de cigarros, que controla a posição de estoque de cigarros, bem como a emissão de notas, relatórios de vendas externas, comissão de vendedores, manutenção de caixa, contas a pagar/receber, tabelas de preços e, ainda, o controle de conta bancária. Tudo isso utilizando um MSX, com 64 Kb de memória. Conheço pessoas que, por não terem um PC em casa, desenvolvem seus trabalhos em um MSX, testam e depois o terminam em um PC.

**CPU - Então, hoje temos uma visão mais madura do MSX?**
Luis Carlos - Sem dúvida alguma, pois o MSX é um ótimo professor. Lamento que nosso pais ainda continue atrasado no que se refere ao sistema MSX.

**CPU - O que você aconselha ao programador de MSX?**
Luis Carlos - Sempre que adquirir um software especifico ou não, converse com colegas para saber se determinado tipo de programa satisfaz suas necessidades ou solicite sua "softhouse", para maiores detalhes.

Na parte de programação, tente fazer o possivel para que seu programa esteja de acordo com o padrão MSX. Por exemplo, no Aquarela, qualquer operação de entrada ou saída para o disk-drive satisfaz qualquer tipo de usuário, seja ele possuidor de uma interface de drive da Sharp, Microsol ou DDX. Seguindo esse raciocínio, seu programa obterá sucesso.

**CPU - O que vocÊ acha de programar para "softhouses"?**
Luis Carlos - O primeiro passo que deve ser dado é o de consultar algumas "softhouses", para verificar o que elas tÊm a oferecer. O programa que você irá desenvolver tem que ter, pelo menos, os seguintes requisitos: ser funcional, inteligente e ter uma apresentação visual semi-profissional, pelo menos.

Se for assinar algum contrato, leia-o antes e pense bem sobre o que leu, pois no meu caso o que aconteceu não foi bem o que eu esperava.

**CPU - Você acha que o programa protegido acaba com a pirataria?**
Luis Carlos - Pelo contrário, a coisa piora. O respeito pelos programas da Nemesis, por exemplo, chega ao ponto dos disquetes possuirem Boots Inteligentes, que não proibem o usuário de ter seu Back Up de segurança.

Por esse motivo, a pirataria em relação a essa empresa é menor. Afirmo isso, me apoiando nas maciças propagandas e também por conhecer muitas pessoas da área.

Alguns usuários que vêem um software travado que impeça a cópia, dão risadas e fazem questão de abrir o software, para espalhá-lo por todo o país. Alguns chegam ao ponto de colocarem no próprio programa o seu nome, para dizerem que foram eles que o destravaram.

O usuário hoje dá valor a um bom programa, que tenha apoio da "softhouse", tanto em dicas como em manuais.

**CPU - O que você fez ou pretende fazer usando um MSX?**
Luis Carlos - O MSX é uma máquina versátil e eficiente, principalmente o MSX 2. Até o momento, além de alguns programas de apoio ao usuário e colegas, desenvolvi o Sistema Gráfico Aquarela, baseado num ponta-pé inicial de um amigo e na colaboração de idéias. Pretendo, dentro de mais algum tempo, retornar com novidades no setor e, nesse periodo, pretendo dar o meu apoio a esta revista, que em muito me ajuda.

**CPU - No momento, o que você desenvolve?**
Luis Carlos - Desenvolvo sistemas comerciais, tanto na linha MSX como na linha PC XT e AT. Troco tecnologia gráfica entre as duas máquinas, no que diz respeito à estética dos menus de controle. Uma das técnicas que costumo utilizar é o recurso de janelas, que dá um toque profissional aos programas e faço questão de fazer igual nas duas máquinas.

**CPU - Na empresa onde trabalha, a Cheyenne, você desenvolve algum trabalho à nivel de Hardware?**
Luis Carlos - O nosso carro-chefe tem sido, até o momento, a Interface Megaram 256K e temos vários outros projetos em estudo.

Achamos que o mercado está bem abastecido e a nossa área, além do desenvolvimento de programas comerciais, também engloba a manutençao de computadores, sejam eles micros, mini ou médios computadores, principalmente a linha MSX e PC.

**CPU - O que mais falaria sobre o sistema MSX?**
Luis Carlos - Volto a repetir que nosso pais está muito atrasado em tecnologia de Informática.

Lá fora, os jovens se divertem com coisas que só vemos em comerciais, como animação gráfica ou músicas digitalizadas, feitas por computadores domésticos modernos. Afinal, até o PC XT 640 Kb já está ultrapassado.

Usuários de computadores são pessoas altamente qualificadas. Podemos lançar programas e periféricos que serão bem aceitos.

# OFERTAS IMBATÍVEIS DA PAULISOFT

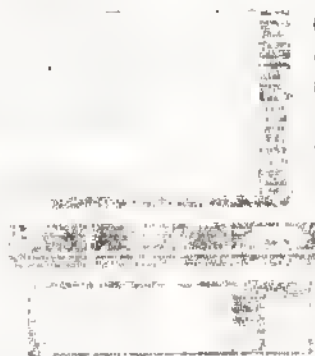
*Conheça as melhores ofertas do mercado.*

## SOFTWARE

- **AQUARELA**: O mais poderoso editor gráfico nacional. Acompanha disco de apoio com mais de 50 alfabetos, diversas molduras e padrões.
- **FAST CDPY**: O copiador mais rápido do mercado. A vergonha dos micros de 16 bits e muitos Kbytes de memória. Comprove.
- **GRAPHIC VIEW**: Genial programa para incrementar suas telas gráficas.
- **MSX TURBD**: Um soft que deixa as rotinas de cálculo de 6 a 20 vezes mais rápidas.
- **EDTRDNIC**: Para montagem e impressão de esquemas para projetos eletrônicos.
- **SPRITE MAKER**: Editor de sprites 16x16 com inúmeras funções.
- **TDP CLI**: Um excenlente programa de cadastro de clientes. Totalmente elaborado em Pascal, o TOP CLI vai atender todas as suas necessidades!
- **APDIDS AQUARELA**: Kit composto de 4 discos de molduras, 4 discos de alfabetos, 1 disco de shapes e 1 disco de padrões e telas.

Todas as novidades em MSX 1.0 e 2.0
Fazemos troca de drives.
Troque seu drive por um mais moderno de 3 1/2 ou 5 1/4 HD 720Kb.
Transformamos a sua TV em um monitor RGB. Consulte.
Promoções de diquetes 5 1/4 e 3 1/2.

**PAULISOFT** INFDRMÁTICA
R. Cel. Xavier de Toledo, 123 - 3º Andar
CEP 01048 - São Paulo-SP
Caixa Postal 2861 - CEP 01051
**Fones: (011)34-5253 E 37-1814**

L&F DESING

Quer comprar, vender ou trocar o seu micro? Procure a Paulisoft que aqui você vai encontrar o melhor negócio. Temos micros da linha MSX, seminovos em excelente estado e com garantia.

Visite nossa loja e comprove

## HARDWARE

- **DISK DRIVE** de 5 1/4" de 40 ou 80 trilhas (360 ou 720 Kb) completo com interface fonte e gabinete. Temos também drives de 3 1/2" (720 Kb).
- **MEGARAM-DISK DDX**: Expansão de memória de 256 Kb para jogos megarom e funciona também como um pseudodrive.
- **KIT 2.D DDX**: Transforme o seu MSX 1.0 para um 2.0 e usufrua de todas as maravilhas de um micro importado.
- **MEGARAM 256 Kb**: Expansão de memória de 256 Kb p/ jogos. OFERTÃO, PRECO IMBATÍVEL.
- **IMPRESSDRA LADY 8D**: 100 CPS. Qualidade carta, totalmente gráfica.
- **FILTRO DE LINHA**: Proteja seu equipamento! 3 tomadas.
- **MDNITDR VITECH CMX/12** Monocromático, 80 colunas.
- **ARQUIVDS** para 100 discos 5 1/4" com chave, em madeira ou plástico.
- **ARQUIVDS** para discos 3 1/2" em plásticos
- **MDUSE INPUT**: Acompanha programa gráfico.
- **MULTI-MDDEM TM2** Gradiente para comunicações micro a micro.
- **CARTÃD DE 8D CDLUNAS** com editor de textos.
- **EXPANSDR DE SLOTS**: com fonte própria. Expande p/ 4 Slots.

## TROCA DE CORRESPONDÊNCIA

Desejo trocar correspondência com usuários do MSX Expert 1.1 e drive 5 1/4".
**Maurício dos Santos**
**Rua Everaldo Marques de Silva, 137 · Jardim Aparecida**
**94800 · Alvorada · RS**

I am a 10 years old girl and I have a 6 years old MSX. My trouble is this: I would like to appeal to your readers for any old games that they don't use anymore or needs they might have on MSX 2. I have had MSX for six years and have always had a terrible time in trying to get tapes once I went a full year without getting a new tape as they are not very popular over here. I would be very grateful for any help.
**Kellyanne Lynn**
**178 Bellfield St. · Dennistown**
**Glasgow G31 IRQ**
**Scotland UK**

Tenho interesse em trocar informações, jogos e programas aplicativos para MSX. Possuo um Expert 1.0 e drive de 5 1/4.
**Décio Baixo Alves**
**Rua Leopoldo Freiberger, 21**
**88160 · Biguaçu · SC**

Troco correspondência com os demais leitores de CPU. Tenho um Expert, drive 5 1/4", gravador e Megaram Disk.
**Alessandro de Silva Oliveira**
**Rua Engenheiro Rebouças, 763 · Caixa Postal 791**
**85800 · Cascavel · PR**

Sendo possuidor de MSX, drive 5 1/4" e Modem, tenho interesse em trocar informações, programas e dicas com outros usuários.

Tenho esquemas das várias versões, algumas dicas e softawre para teste do hardware, em listagem, do Multimodem Telcom.
**Marcelo Silva**
**Rua Mucio Teixeira, 221 ap 202**
**90080 · Porto Alegre · RS**

Troco programas para MSX 2.0, com ou sem Megaram.
**Mauricio Faria Dome Menzano**
**Rua Professora Zilda Andrade, 197/103 · Bairro de Lourdes**
**29000 · Vitória · ES**

## SUGESTÕES E DÚVIDAS

Como posso acessar, em Turbo Pascal 3.0 MSX os caracteres (em hexa) 10 a 1B? Já tentei utilizar chr(x) e ^x, sem sucesso. Os outros caracteres funcionam normalmente. Existe alguma outra versão de Pascal para MSX?

Quais as versões existentes de Cobol e C para MSX?
**Marcos José Pinto**
**Caixa Postal 91395**
**25600 · Petropólis · RJ**

*Para acessar os caracteres 10H a 1BH ou qualquer caracter abaixo de 1FH, inclusive, basta proceder como em BASIC. Esses são caracteres de controle, sendo necessário informar à BIOS que queremos que sejam impressos e não interpretados. Isso é feito enviando o caracter 01H antes do código adicionado de 40H (64D), ou seja, para imprimir (na tela) o caracter 10H use o seguinte comando PASCAL:*

```
write(CHR(1),CHR(16+64)); (* 10H = 64D *)
```

*Os compiladores COBOL e C disponíveis para o MSX são os mesmos que estão disponíveis para CPM. Um compilador C muito popular é o Aztec C, disponível em disco.*

Inicialmente quero lhes dar os parabéns pela excelente revista que publicam. Seu conteúdo é excepcional. Sou estudante de processamento de dados e muitos colegas meus possuem esta maravilhosa máquina de 8 bits, o MSX, e a todos eles faço questão de apresentar a revista CPU com muito orgulho.

Mas, vamos aos meus problemas:

Aqui em São Paulo a revista apareceu apenas do número 6 em diante e possuo os números 6 e do oito em diante, sendo que me falta o número 7, na qual foi iniciado o projeto MSXDEBUG. Acho o utilitário super interessante, pois além de sua utilidade, ele nos ajuda a desvendar os segredos dessa máquina. Gostaria, então, de poder recebê-la através de reembolso postal.

Digitei o projeto SCREEN IV até a parte que foi publicada no número 10, porém estou enfrentando um sério problema. Não consigo gravá-lo no endereço indicado (&H4100 em diante). O micro acusa

função ilegal. Estou gravando-o em outro endereço, de A100H em diante, mas quando tento executá-lo, através do DOS, a máquina proporciona um KEYOFF, não sai do DOS, trava o teclado e fica acessando o drive direto. Não sei se é por causa do endereço de gravação. Se for, como fazer para gravar no endereço correto, ou seja, de 4100H em diante?

Estou usando um bom monitor para digitar, que transfere os blocos de memória digitados para o endereço correto mas, ao usar o comando BSAVE"SCREEN.COM",&H4100,etc, do Basic, o computador não aceita. O que devo fazer?

E quanto ao problema da execução? Já revisei a digitação e ela está correta.

**José Lui Inácio**
**Rua Jacinto Lima Santos, 269 · Vila Libaneza**
**03738 · São Paulo · SP**

*Observe que o SCREEN IV deve ser digitado utilizando algum utilitário que seja capaz de salvar o programa em disco, sem alterar nem acrescentar nada ao código original. Salvando o código a partir do BASIC, com o comando BSAVE, o mesmo fica acrescido de 7 BYTES, o que faz com que os desvios e chamadas fiquem deslocadas.*

*Procure adquirir um utilitário que funcione como indicado. Aliás, o MSXDEBUG é o utilitário que está sendo usado pela revista para a digitação do SCREEN IV.*

*Caso deseje que efetuemos a gravação dos projetos MSXDEBUG e SCREEN IV, até a parte que foi publicada na revista, envie-nos um disco e a quantia de Cr$ 150,00, para cobrir as despesas postais, que lhe devolveremos o disco com os programas gravados.*

Estou escrevendo esta carta para saber como terminar com o problema que tenho em relação ao projeto MSXDEBUG.

Digitei o bloco 1 da rotina que implementa o comando DASS, verifiquei a soma e fiz o reconhecimento da rotina, usando o seguinte comando: EXEC 105A 100.

Apareceu o que era esperado, ou seja, a posição da memória (a partir do 100), uma sequência de BYTES e a instrução em Assembler correspondente. Pressionei ENTER e os próximos códigos foram aparecendo, pressionei a barra-de-espaço, houve um CLS, e as próximas 24 instruções foram mostradas. Depois, troquei as teclas de função, como sugerido. Troquei a versão para 1.5. Retornei ao DOS com a nova versão gravada no disco e a executei.

Até este ponto tudo correu bem. Digitei o comando DASS 100 e DASS 4100 100, sendo que ambas as instruções acusaram erro de comando existente.

Carreguei minha nova versão no endereço 4100H e executei o comando DISP 4D08, encontrando o BSAVE como último comando do MSXDEBUG. Então, escrevi DASS logo após o BSAVE, separando-os com o Byte 00H e depois do DASS escrevi como estava antes, isto é, o BYTE 00H logo após o DASS e depois os BYTES FFH e 1AH. Gravei de novo e executei. Testei o comando DASS o computador tirou o MSXDEBUG do ar e tornou a voltar ao DOS.

Gostaria que o autor do projeto MSXDEBUG solucionasse o meu problema.

**Alexandre Nogueira Catelli**
**Rua Senador Vergueiro, 200/401 · Flamengo**
**22241 · Rio de Janeiro · RJ**

*Seu problema pode ser facilmente solucionado. Segundo seu relato, todos os passos foram seguidos corretamente, exceto com relação ao endereço de definição de entrada da rotina. Sem esse detalhe, apesar do comando ser reconhecido, sem apresentar mensagem de erro, o controle do sistema é transferido para algum ponto qualquer da memória.*

*Observe que a solução está no artifício utilizado pelo leitor para testar a rotina. Verificado seu funcionamento, restava saber se a entrada, além do comando, foi corretamente definida.*

Sempre que se pensa em MSX, lembra-se do Japão e da Europa. Porque o MSX não existe nos EUA?

Queria saber como vai o padrão MSX no resto do mundo e acho que deveriam fazer uma comparação entre os micros MSX 2.0, o Amiga 500 e o Atari ST 1040, citando os principais periféricos disponíveis no exterior e os respectivos preços.

Comentem o que ocorre de novo lá fora em termos de MSX, para sabermos o que poderá aparecer aqui.

Teoricamente, é possível a instalação de um winchester no MSX?

Existe algum hardware que torne o MSX compatível com IBM PC? O Apple II plus e o Atari ST 1040 possuem placas para esta finalidade.

Gostaria de saber como posso ligar o meu MSX a um sintetizador (no caso um DX7), que tenha interface MIDI.

Um fato que venho observando é que, apesar de muitas pessoas possuirem drives de 3,5" e a Gradiente ter lançado o seu micro com esse tipo de drive, existem poucos aplicativos que funcionem corretamente nesse "novo" formato. Entre eles, cito o Hello, Graphos 3, Fastcopy, e o Copygts.

**Valdir Souza Junior**
**Rua Abel Capela, 481 · Coqueiros**
**88080 · Florianopólis · SC**

*Estamos providenciando uma matéria de comparação entre vários padrões de micros, onde abordaremos as características técnicas de cada padrão e preços de periféricos.*

*O mercado MSX, lá fora, pelas notícias que têm chegado, indicam que o padrão vai indo bem e que brevemente será lançado o MSX 3.*

*É possível ligar um winchester no MSX, sendo que lá fora existem interfaces para esse propósito. Para tornar um MSX parecido com um IBM PC ainda não existe nada, mas parece que o MSX 3 trará algumas novidades nesse sentido.*

*Para ligar um MSX a um sintetizador dotado de MIDI, basta colocar uma interface MIDI no seu MSX. Já existe no Brasil esse tipo de interface para o MSX, fabricada pela Blump Robótica, empresa de São Paulo.*

*A utilização de drives de 3,5" está se tornando cada vez comum e as softhouses devem começar a adaptar seus programas para funcionarem também neste tipo de equipamento, pois a demanda será cada vez maior.*

Quando tento carregar o jogo Robocop em meu micro Expert Plus, com drive de 5 1/4 e interface DDX última versão, após o programa solicitar que o drive seja ligado, nada mais acontece e nem mesmo o drive é acionado.

**Marco Antonio**
**Rua 24 de Outubro, 24**
**07060 · São Paulo · SP**

*A cópia do jogo que possui está com defeito, pois o jogo Robocop roda nos novos micros da Gradiente. Entre em contato com a softhouse que lhe vendeu o programa e solicite a troca de sua cópia por uma que funcione.*

Li o artigo a respeito da Megaram e gostaria de obter algumas informações. Comprei uma Megaram da marca Cheyenne e alguns jogos, como Nemesis I e F1 Spirit não rodam.

Já troquei três vezes e o problema não foi solucionado. Um colega meu emprestou-me a sua Megaram, da mesma marca, e funcionou perfeitamente com todos os jogos.

Gostaria de saber qual o comando que devo usar no Cobol para limpar a tela, pois utilizei o comando ERASE do PC e não funcionou.

Tenho interesse em trocar correspondência com os demais leitores de CPU.

**Ivam Andrucholli**
**R. Gen C. Lobo Botelho, 92 · Vila Maria**
**02169 · São Paulo · SP**

*Acredito que o problema que está tendo com a sua Megaram só pode ser resolvido com a ajuda do fabricante. Entre em contato com ele e solicite que ele cumpra com suas obrigações.*

*Estamos publicando uma série de artigos sobre a linguagem Cobol, que irão lhe ser úteis, não só para a dúvida que nos apresentou como para outras que irá ter quando começar a utilizar a linguagem Cobol no MSX.*

Gostaria que falassem um pouco sobre o comando CALL. Para que serve, como se usa, etc. Também poderiam publicar alguma matéria sobre o DOS e seus recursos, pois tenho muitas dúvidas em relação ao seu uso.

Quem gostar de jogos do tipo adventure, como eu, entre em contato, para que possamos trocar maiores informações.

**Rodrigo M. Valladão**
**Av. Ary Parreiras, 74/304**
**24230 · Niteroi · RJ**

## DICAS

Primeiramente, queria parabenizá-los

pela revista que editam.

Eu e meus amigos temos um MSX Clube, a Mirage Software, e gostaria de entrar em contato com outros usuários de MSX.

Gostaria de saber porque jogos como Robocop, Double Dragon e Thunder Blade não rodam num Expert Plus.

Aproveito para enviar algumas dicas:

Para desativar o comando LIST, digite POKE &hFF89,&hE1

Para destaivar o CONTROl+STOP, digite POKE &HFBB1,0

No jogo TETRIS, para entrar no modo de demonstração com música, após carregar o jogo, pressione o cursor para cima e a barra de espaço.

**Fernando M Hemtascke**
**Av. Trompowsky, 50/23 · Ed. Fernanda**
**88015 · Florianópolis · SC**

Envio algumas dicas, para que outros usuários também possam terminar os jogos.

F1 SPIRIT - Senha: Usar a opção "COMMAND" e vá para a opção "INPUT PASSWORD", entrando com a seguinte senha:
NGLJDDILEBFLKJEIIAPLIBED.

Com essa senha o jogo já começa com todas as pistas disponíveis e, caso você vença a última pista de F1, você irá entrar direto no "MEMORIAL OF FORMULA 1".

LA HERANCIA - Senha: OLAAGAKA, que serve para qualquer fase.

PARODIUS - Todas as armas - Tecle <F1>e digite ZENBU, seguido de <RETURN>

GAME OVER 2 - Senha - Digite 65535 para iniciar o jogo.

RISE OUT · Mais vidas - tecle <CONTROL, <SHIFT> e 2

METAL GEAR - Ao iniciar, tecle <F1> e entre com uma das duas senhas a seguir: DS4 para começar com 4 estrelas ou HIRAKE GOMA para começar com 8 cartões.

**Carlos Henrique Pessoa de Menezes e Silva**
**Rua Odete Oliveira Lacourt, 1226/203**
**Jardim da Penha**
**29060 · Vitória · ES**

## ELITE

Possuo uma gravação do jogo Elite com cerca de 2 milhões de créditos, sendo que o piloto está na posição de COMPETENT e a nave com uma série de equipamentos.

**Rafael Stoll Martins Machado**
**Rua Huet Bacelar, 42 · Jardim Sabará**
**91300 · Porto Alegre · RS**

## LA HERANCIA

Na edição número 13 foi publicado uma matéria sobre o jogo "La Herancia", por Leonardo Letelier.

Pelo artigo, na fase I, seria suicídio entrarmos no táxi sem o passaporte.

Mesmo se entrarmos no táxi sem o passaporte e, na fase II dermos dinheiro para o mendigo, este nos dará nosso passaporte.

Na fase II, temos que entrar no ônibus número 9, mas tem que ser o segundo ônibus número 9.

**Ricardo Mecelis**
**Av. Oswaldo Cruz, 199/1405 · Flamengo**
**22250 · Rio de Janeiro · RJ**

## OPINIÃO

Gostaria de manifestar meu apoio ao leitor Fernando Barros Maylinch, cuja carta foi publicada em CPU número 14.

O MSX 2+ é muito mais barato que o Amiga, além de ter algumas vantagens, como 14 canais de som, contra os 4 do Amiga, 19268 cores contra as 4096 do Amiga. O MSX é o micro mais vendido e tem o maior número de unidades instaladas no país líder na economia, o Japão. Se não há MSX nos EUA é porque eles não querem mais artigos japoneses atrapalhando o decadente mercado deles que, aliás, já está repleto de marcas japonesas assumindo o primeiro lugar em vendas.

O dia em que as maravilhosas marcas japonesas invadirem o Brasil, muitas marcas nacionais, que faltam com o apoio e respeito ao usuário, terão que melhorar muito.

**Eduardo Loos**
**Calxa Postal 78**
**88350 · Brusque · SC**

# POR DENTRO DA INTERFACE DE DRIVE

PARTE III

Júlio Veloso

## ERRATAS

No artigo publicado anteriormente, houve um erro no programa EXAFAT. Pelo fato de tal erro ser aparentemente imperceptível (pois somente em alguns tipos de FAT ele é notado), não pude dectetá-lo na época da elaboração da matéria. Trago aqui as modificações que devem ser feitas para solucionar o problema (listagem 6).

Também no número anterior dois pequenos erros foram notados na página 10, onde aparece o resultado da compactação — falta um byte 00 no endereço 11. Esse erro me influenciou na hora de contar o número de bytes economizados, que na verdade são 4 em vez de 3 bytes.

A expressão para se encontrar o endereço de uma chave qualquer pode ser reduzida matematicamente para (4*N-3). Tal procedimento, contudo, não é aconselhável. Caso se queira uma flexibilização no tamanho dos registros, poderíamos escrever a expressão com ((N-1)*R+1), onde R é o tamanho físico da cada registro.

Na primeira parte da série destes artigos, o parâmetro de entrada C para a rotina PHYDIO saiu como C = 1111 1T0F b, pois eu não sabia da existência de drives com formatações do tipo 8 setores/trilhas e que utilizam a mesma rotina.

O parâmetro, na verdade, ficaria da seguinte forma:

C — 1111 1TSF B sendo:
S — 0 - 9 setores/trilha
1 — 8 setores/trilha

Deste modo, a rotina fica generalizada para todos os tipos de drives existentes, embora os que utilizam 8 setores/trilhas não sejam utilizados no MSX.

## INTRODUÇÃO

O resultado do trabalho de "Debugação" ao qual me referi no artigo passado, resultou no surgimento de informações importantes, como as variáveis da interface e a solução de alguns de seus "bug's".

No trabalho de identificação das variáveis, contei com a ajuda de meu colega Marcelo Temer. Caso outros leitores disponham de outras variáveis que não estejam publicadas neste artigo, envie para a redação de CPU, para que possamos complementar a tabela.

No artigo anterior desta série, fiz um anúncio sucinto informando que mostraria duas outras importantes rotinas sobre Drive. São elas: SAVE e LOAD, que vêm a ser as rotinas de gravação e carregamento de arquivos do disco. Neste artigo só apresentarei as rotinas de carregamento, deixando as de gravação para depois.

## A RESPEITO DA EXAFAT

A versão publicada no número anterior é para os usuários que utilizam o cartão de 80 colunas. A listagem em Basic desse número destina-se aos usuários que ainda não trabalham com este periférico.

Para a digitação, a parte em Assembler é idêntica ao do número anterior, sendo que só a parte escrita em Basic é que necessita ser alterada, conforme a listagem 7 publicada, a partir da linha 1000.

A linha 1590 da listagem é responsável pelo help de tela. Por ser a tela pequena, os dados de alguns arquivos não podem ser impressos por inteiro. Por isto, caso algum leitor não queira este help, basta retirar esta linha do programa, sem se preocupar que ocorra algum problema.

Para executar o programa EXAFAT, deve-se executar o comando RUN 1000 e, para testar o seu funcionamento (verificar se o conteúdo dos clusters mostrados corresponde com o que foi gravado), basta executar o programa que saiu antes da listagem do EXAFAT. Para isto digite o comando RUN.

Para que o programa teste do EXAFAT publicado no artigo anterior funcione para usuários que não possuem a placa, basta modificar a linha 180 para: 180 FOR I=1 to 8.

## VARIÁVEIS DO BDOS

Neste artigo se encontra uma tabela de variáveis, a tabela 1, resultado do trabalho de debug da interface. Não achei interessante publicar o funcionamento e os parâmetros de entrada e saída da BDOS, bem com as variáveis ligadas ao processo de leitura e gravação dos arquivos, pois estou fazendo todo o trabalho de construção da simulação do BDOS. Junto às variáveis, temos alguns endereços de rotinas úteis ao usuário.

Não foi possível identificar todas as variáveis e rotinas do BDOS, mas as importantes estão descritas.

No quadro, existem vários termos que, por serem muito específicos, podem gerar alguma dificuldade de interpretação por parte do leitor. A estrutura destes termos vem a seguir:

— Cada disco tem sua DBP (Disc Bloc Parameter). Se você tem dois drives, terá um DBP para o drive A e outro para o drive B, mostrando as características ou parâmentros de acesso deste disco. Neste artigo, trago a estrutura do DBP (tabela 2).

| Endereço | Descrição da Variável ou rotina |
|---|---|
| 0001 / 0002 | Endereço da rotina que chama o DOS |
| 0006 / 0007 | Endereço da rotina que chama as rotinas do BDOS |
| 0031 / 0002 | Endereço do RST 30h para chamar rotinas de outros Slot's |
| C200 | Entrada do DOS |
| D60A / D70A | Buffer de teclado usado no DOS |
| D713 | Indica ao DOS que existe um arquivo na memória |
| D714 | Indica que o DOS está na memória |
| D717 / D718 | DMA |
| D71B / D71C | Endereço da entrada do DOS |
| D71D | Flag para indicar ao DOS arquivo Batch em processamento |
| D93E á D963 | FCB usado para carregar programas do disco no DOS |
| D940 á DA33 | Área de mensagens de erro |
| DA34 á DA70 | Tabela de endereços para as rotinas do BDOS |
| DC66 á DC8B | FCB para carregamento do SOLXDOS |
| DF94 á F194 | buffer's de disco e DPB |
| F195 á F1A9 | DPB drive A |
| F1AA á F1BE | DPB drive B |
| F1BF | Delay drive geral (120 á 0) |
| F1C0 | Delay drive A (60 á 0) |
| F1C1 | Delay drive B (60 á 0) |
| F1C2 | Nº do último drive usado |
| F1C3 | área livre |
| F1C4 | área livre |
| F1C5 | área livre |
| F1C6 | área livre |
| F1C7 | área livre |
| F1C8 | Número de drives conectado |
| F1C9 á F1DB | Parte da rotina de impressão |
| F1D9 á F1E1 | Rotina de cópia de bloco |
| F1E2 á F1E7 | Rotina de chamada do DOS |
| F1EB á F1F3 | Rotina de chamada do conteúdo de HL |
| F1F7 á F209 | Tabela de periféricos para file-name |
| F23B | Eco para impressora (0- não faz eco / 1- faz eco) |
| F23D / F23E | Endereço do DMA |
| F243 / F244 | Início da área do DPB |
| F247 | Número de drives ativos do sistema |
| F248 | Dia |
| F249 | Mes |
| F24A | Ano (+ 1980) |
| F24F á F266 | Ganchos Nº 1 (para 21 rotinas) |
| F267 / F268 | Mapa de locação (FAT) |
| F26A / F26B | Mapa de locação (FAT) |
| F2AF | Uso da rotina 2 (Envia um caracter ao console) |
| F2B9 á F2C3 | Nome do arquivo |
| F2C4 | Drive origem |
| F2C5 á F2CF | Nome novo |
| F2D0 | Drive destino |
| F2D1 á D2DB | Nome antigo |
| F304 / F305 | Endereço do stack |
| F306 | Slot da interface |
| F307 / F308 | Endereço do FCB para a rotina 12 h |
| F309 / F30A | DPB Defaut |
| F323 / F324 | Endereço do apontador para a rotina de erro |
| F325 / F326 | Endereço do apontador para a rotina de fim de arquivo BATCH |

TABELA 1 — Variáveis e rotinas ligadas ao BDOS

| Endereço | Descrição da Variável ou rotina |
|---|---|
| F327 á F32B | Gancho da rotina BDOS 3 |
| F32C á F330 | Gancho da rotina BDOS 4 |
| F331 á F335 | Gancho da rotina inicialização BDOS |
| F336 | IOBYTE controle (3: CTRL + STOP, FF: IOBYTE cheio, 00: IOBYTE vazio) |
| F337 | byte lido do console |
| F341 | Número do slot da Página 0 - RAM |
| F342 | Número do slot da Página 1 - RAM |
| F343 | Número do slot da Página 2 - RAM |
| F344 | Número do slot da Página 3 - RAM |
| F346 | Indica se o DOS foi carregado antes do BASIC DE DISCO (FF h) |
| F347 | Número de drives físicos do sistema (<control> na partida) |
| F348 | Indica em qual slot a interface de drive está conectada |
| F349 / F34A | Endereço da FAT |
| F34B / F34C | Endereço de uma área de variáveis |
| F34D / F34E | Endereço da FAT |
| F351 / F352 | Endereço do buffer de setores |
| F353 / F354 | Endereço de uma rotina em L.M. |
| F355 á F364 | F195/F1AA/0000/0000/0000/0000/0000/0000 - Vetores dos DPB's |
| F365 á F367 | Rotina que lê a porta (A8) |
| F368 á F36A | Rotina que abilita BDOS na página 1 (end: 4000h) |
| F36B á F36D | Rotina que abilita RAM na página 1 (end: 4000h) |
| F36E á F370 | LDIR Página 1 da ram |
| F371 á F373 | chama rotina BDOS 3 |
| F374 á F376 | chama rotina BDOS 4 |
| F377 á F379 | JP 0 |
| F37A á F37C | JP 0 |
| F37D á F37F | chama rotina que chama o BDOS |
| F380 á F382 | pega dado da memória = D: porta de memória E: valor lido |
| F385 á F387 | escreve dado na memmória = E: valor D: porta de memória |
| F38C á F38E | chama o endereço em IX |

— A partir do endereço de memória F355H até F364H, se encontram 8 vetores que servem para indicar os DPB's dos discos dos drives conectados. Caso se queira saber onde começa o DPB do drive A, devemos verificar os conteúdos dos endereços F355H e F356H, montando um endereço. Para o drive B, temos os endereços F357H e F358H, e asim sucessivamente.

— DMA significa, como foi dito no primeiro artigo desta série, Disc Memory Adress, e é o ponteiro onde o BDOS colocará o bloco a ser lido ou onde deverá ter o bloco para gravação.

— De F1BFH a F1C1H encontramos os Delays, ou seja, o tempo gasto, usado pela rotina de interrupt, até fazer parar o drive por completo.

## ERROS DO BDOS

Um dos erros que notei refere-se à volta ao DOS. Esta operação atualiza um conjunto de variáveis e, ao fazer isto, altera também a FAT. Quando há uma troca de disco e não se avisa ao DOS que houve esta troca, este grava a FAT antiga em cima da nova.

Para solucionar este problema, encontrei uma rotina dentro da rotina de 'boot' do DOS, que faz a alteração de algumas variáveis, impedindo que essa atualização seja feita e que uma troca de disco não resulte na perda de sua FAT. A rotina tem o nome de ALTVAR e deve ser incluída antes de chamar o DOS (ver listagem 4).

Outro grande problema do BDOS (da interface Microsol) é no tratamento de erro. Não chega bem a ser um erro, mas uma maneira que não considero certa de tratar o problema, pois impede que programas que trabalhem com a tela cheia processem este erro de uma forma controlada, resultando em uma tela borrada e sem controle.

A solução para este problema foi encontrada também durante o 'debugamen-

to' da interface. O BDOS utiliza a variável F323H como um ponteiro para um vetor que indica a rotina que trata este erro, utilizando para todos os erros fatais uma frase que instrui o usuário sobre como proceder para abortar, repetir ou ignorar uma operação.

Essa variável, se alterada para #72AE, faz o procedimento de erro funcionar no Basic. Caso se queira alterar para uma rotina própria, é só estudar a rotina genérica (listagem 3) e encaixar a sua.

## AS ROTINAS DE CARREGAMENTO DE ARQUIVO

Como o próprio título diz, essas são as rotinas de auxílio no carregamento de um arquivo do disco, que utilizam as rotinas publicadas no número anterior e que fazem a tradução da FAT (SQCONV) e a rotina de carregamento de setor (PHYDIO).

Junto a elas são publicadas outras rotinas de apoio que simulam as rotinas do BDOS. Elas são similares para se manter um padrão com os programas já feitos.

A única modificação que dever ser feita para se utilizar a rotina ~~LOAD~~ READ, como se fosse para o BDOS convencional, é trocar a linha:

LD DE,(TAMSET), por LD DE, #80

pois o BDOS convencional utiliza um buffer de tamanho de 128 bytes, em vez de um cluster que tem o tamanho usado na rotina READ.

As rotinas READ, OPEN e ADMA devem ser modificadas pelas micro rotinas dispostas na listagem 1.

Não adiantaria simular o BDOS se não criássemos melhorias. Estas são: interpretação da FAT uma vez apenas, verificação e gravação do atributo de um arquivo, possibilidade de criar labels (rótulo de disco), obtenção do cluster atual, a movimentação desse pelo arquivo, etc.

A respeito do FCB e tratamento de arquivos das rotinas de simulação, temos:

— o FCB é diferente do utilizado pelo BDOS convencional, pois utiliza os 25 bytes de espaço livre depois do nome de forma diferente;

— é possível abrir e ter um controle de vários arquivos ao mesmo tempo (o número varia de acordo com a saturação do diretório);

— existe um controle interno para verificação do atributo de arquivo. A saída no registrador 'A' da rotina READ e OPEN se refere a este atributo.

| Pos | Descrição |
|---|---|
| 00 | Nº do drive (0=A,1=B,2=C etc) |
| 01 | Formatação (F8/F9/FA/FB/FC/FD/FE/FF) |
| 02/03 | Bytes por setor |
| 04 | (0F h) |
| 05 | Número de drives físicos da interface |
| 06 | Entradas no diretório (0-64 / 1-112) |
| 07 | Setores por cluster |
| 08/09 | Setor inicial do disco (local da FAT) |
| 10 | Número de FAT's |
| 11 | Entradas no diretório |
| 12/13 | Início da área livre (setor) |
| 14/15 | Cluster's utilisáveis |
| 16 | Setores por FAT |
| 17/18 | Início do diretório (setor) |
| 19/20 | Endereço da FAT (na área de variáveis) |

TABELA 2 — DPB (Disc Parametrer Block)

```
OPEN   :PUSH DE
        LD C,#F
        CALL BDOS
        DEC A
        CCF
        POP DE
        RET

READ   :PUSH HL
        PUSH DE
        PUSH BC
        LD C,#14
        CALL BDOS
        POP BC
        POP DE
        POP HL
        DEC A
        CCF
        RET

ADMA   :LD C,#1A
        CALL BDOS
        RET
```

LISTAGEM 1 — Micro-rotinas para leitura de arquivos pelo BDOS convencional

```
ROT1    :JP STCONV
ROT2    :XOR A
         JP TRQDSC
ROT3    :JP SQCONV
ROT4    :JP MNTDIR
ROT5    :JP ERRMAP
ROT6    :JP CACFAT
ROT7    :JP ATLMAP
ROT8    :JP SAVFAT
ROT9    :JP GRVNUM

TAB1    :DEFW BUFPGM
TAB2    :DEFW BUFDIR
TAB3    :DEFW DIRET
TAB4    :DEFW BUFMAP
TAB5    :DEFW BAKFAT

ROT10   :JP LOAD
ROT11   :JP OPEN
ROT12   :JP READ
ROT13   :JP CPESC
ROT14   :JP ADMA
ROT15   :JP AVCSEQ
ROT16   :JP RETSEQ
ROT17   :JP MSTCLR
ROT18   :JP MSTTAM
ROT19   :JP MSTDAT
```

LISTAGEM 2 — Alteração da tabela de JUMP'S

```
;
;                  ROTINA DE TRATAMENTO DE ERROS FATAIS
;
;
; Ponto de Entrada: 1º elemento (endereço) da tabela de endereçada por #F323
;
; Entrada: C - tipo de erro: 0   - Erro de proteção
;                            2   - Drive inativo
;                        bit 0: 1- Erro de escrita, 0- Erro de leitura
;          A - drive onde ocorreu o erro
;
; Saída:   C - Opção do usuário: 0 - Ignorar,
;                                1 - Tentar de novo,
;                                2 - Abortar
;
ROTERR:ADD A,"A"     ; >coloca nas frases o drive onde ocorreu o erro (X:)
       LD (ERRM1+28),A
       LD (ERRM4+27),A
       LD (ERRM5+8),A
       LD A,C
       AND #FE       ; >mascara para isolar os bit's de 1 á 7 do acumulador
       OR A          ; >testa se é zero para erro de proteção
       LD DE,ERRM4
       JR Z,ROTE02
       CP #02        ; >testa se é um para erro de drive inativo
       LD DE,ERRM5
       JR Z,ROTE02
       LD HL,ERRM2   ; >HL para erro de LEITURA
       BIT 0,C       ; >testa bit 0 para ver se é erro de leitura
       LD BC,8
       JR NZ,ROTE01
       LD HL,ERRM3   ; >altera HL caso seja erro de GRAVAÇÃO
ROTE01 LD DE,ERRM1+10
       LDIR          ; >transfere a palavra 'LEITURA' ou 'GRAVAÇÃO'
       LD DE,ERRM1   ; >inicio do erro de leitura ou gravação
ROTE02 CALL #F1C9    ; >imprime o erro ocorrido
ROTE03 LD DE,ERRM6
       CALL #F1C9    ; >imprime as opções para o usuário
       CALL #5445    ; >rotina de teclado
       PUSH AF
       CALL #5183    ; >rotina que imprime a tecla precionada (CHPUT)
       POP AF
       AND #5F       ; >converte minuscula em maiúscula
       LD C,0        ; >atribui C=0 p/ "I", C=1 p/ "T" e C=2 p/ "A" dependen-
       CP "I"        ;  da tecla precionada.
       RET Z
       INC C
       CP "T"
       RET Z
       INC C
       CP "A"
       RET Z
       JR ROTE03

ERRM1 :EQU #D940    ; '  Erro de GRAVAÇÃO no drive X:'
ERRM2 :EQU #D95F    ; 'GRAVAÇÃO'
ERRM3 :EQU #D967    ; 'LEITURA '
ERRM4 :EQU #D96F    ; '  Disco protegido no drive X:'
ERRM5 :EQU #D98D    ; '  Drive X: inativo'
ERRM6 :EQU #D9A0    ; '(A)boortar, (I)gnorar ou (T)entar de novo'
```

**LISTAGEM 3 — Rotina da tratamento de erro retirado de RAM do MSX**

Neste número apresento 9 destas rotinas, que funcionam da seguinte forma:

**LOAD** — carrega um arquivo na memória

Entrada:
DE — endereço do FCB
HL — endereço inicial
Saida
NC — arquivo lido
C — erro de leitura

**OPEN** — abre um arquivo para leitura — similar ao BDOS

Entrada:
DE — Endereço do FCB
Saida:
NC — arquivo aberto
C — erro de operação
A — FF : sem entrada
FE : sem FAT
02 : arquivo protegido(por atributo)
00 : operação ok

**READ** — lê um cluster do arquivo aberto (similar ao BDOS).

Entrada:
DE — endereço do FCB
Saida:
NC — cluster lido
C — erro de leitura
A — FF : erro de leitura
02 : arquivo protegido (por atributo)
01 : fim do arquivo
00 : operação ok

**CPESC** — compara a tecla ESC para interromper a leitura do arquivo.

Saida:
F: — NC : ESC pressionado
C : ESC não pressionado

**ADMA** — ajusta DMA (Disc Memory Adress) — similar ao BDOS

Entrada:
DE — endereço de memória do buffer usado para transferir dados da memória para o disco e vice-versa.

**AVCSEQ** — avança um cluster no ponterio de arquivo

Entrada:
DE — endereço do FCB
Saida:
Flag Z — fim do arquivo
Flag NZ — operação ok

**RETSEQ** — retorna um cluster no ponteiro de arquivo

```
ALTVAR  :LD HL,#F355
         LD A,(#F347)
ALTV01  :LD E,(HL)
         INC HL
         LD D,(HL)
         INC HL
         PUSH HL
         PUSH DE
         POP IX
         LD H,(IX+#14)
         LD L,(IX+#13)
         DEC HL
         LD (HL),0
         POP HL
         DEC A
         JR NZ,ALTV01
         RET
```

**LISTAGEM 4 — Rotina para solução do erro de retorno ao DOS**

Entrada:
DE — endereço do FCB
Saida:
Flag Z — fim do arquivo
Flag NZ — operação ok

**MSTCLR** — mostra o número do cluster final

Entrada:
DE — endereço do FCB
Saida:
HL — valor do cluster atual

**MSTTAM** — mostra o tamanho do arquivo aberto (similar ao BDOS).

Entrada:
DE — endereço do FCB
Saida:
HL,DE — tamanho do arquivo

**MSTDAT** — mostra a data do arquivo aberto (similar ao BDOS).

Entrada
DE — endereço do FCB
Saida:
HL — data codificada

Para que as novas rotinas funcionem adequadamente, inclua-as na listagem 5 deste artigo depois da listagem 3 (fonte do assembler) do número anterior do número anterior, alterando também a tabela de jumps, conforme a listagem 2.

Nos próximos números, descreverei o processo de gravação de arquivos e o restante da simulação do BDOS, incluindo novas rotinas de atributo, label, subdiretórios, etc.

```
LOAD   :PUSH DE                    ; << LÊ UM ARQUIVO DO DISCO >>
        PUSH HL
        CALL OPEN     ; > abre o arquivo para leitura
        POP DE
        CALL ADMA     ; > indica o endereço inicial de carregamento
        POP DE
        RET C
LOAD1  :PUSH DE
        CALL READ     ; > lê um cluster do arquivo
        POP DE
        RET C                      ; > erro de leitura?
        RET NZ                     ; > final de arquivo
        JR LOAD1

OPEN   :PUSH DE                    ; << ABRE UM ARQUIVO PARA LEITURA >>
        LD A,(DE)
        CALL CONVDRV
        LD DE,DIRET   ; > procura nome no diretório
        POP HL
OPEN1  :PUSH HL
        PUSH DE
        LD B,11
        INC HL
OPEN2  :LD A,(DE)
        CP (HL)
        JR NZ,OPEN3   ; > se for diferente ..
        INC HL
```
LISTAGEM 5

```
        INC DE
        DJNZ OPEN2
        POP DE                          ; > se for igual
        POP HL
        JR OPEN5
OPEN3  :POP HL                          ; > avança um nome no diretório
        LD DE,32
        ADD HL,DE
        EX DE,HL
        POP DE    POP HL
        PUSH HL
        PUSH DE
        LD HL,DIRET   ; > acha o endereço final do diretório
        LD DE,#200
        LD A,(FORMAT2+1)
        LD B,A
OPEN4  :ADD HL,DE
        DJNZ OPEN4
        POP DE
        CALL CPHLDE   ; > verifica fim de diretório
        POP HL
        LD A,#FF
        JP Z,ERRGER
        LD A,(HL)   LD A,(DE)
        AND A
        JR NZ,OPEN1
        LD A,#FD
        JP ERRGER
OPEN5  :LD (POSNOM),DE                  ; > pega Tamanho e Entrada
        PUSH DE
        POP IY
        LD A,(IY+11)
        LD (ATRARQ),A
        CP 2
        JP Z,ERRGER   ; > verifica se o aquivo pode ser lido
        PUSH HL
        LD L,(IY+28)
        LD H,(IY+29)
        LD (LOWTAM),HL
        LD L,(IY+30)
        LD H,(IY+31)
        LD (HITAM),HL
        LD H,(IY+27)
        LD L,(IY+26)
        LD (ENTARQ),HL
        LD H,(IY+25)
        LD L,(IY+24)
        LD (DATARQ),HL
        LD IY,BUFPGM+2                  ; > procura trilha do arquivo a ser carregado
        LD H,(IY+1)
        LD L,(IY+0)
OPEN6  :LD DE,(ENTARQ)
        CALL CPHLDE   ; > verifica se é a entrada correta
        JR Z,OPEN8
OPEN7  :INC IY
        INC IY
```

```
        LD H,(IY+1)
        LD L,(IY+0)
        LD DE,#FFF
        CALL CPHLDE   ; ) encontra próxima entrada
        JR NZ,OPEN7
        INC IY
        INC IY
        LD H,(IY+1)
        LD L,(IY+0)
        LD DE,#FFFF
        CALL CPHLDE   ; ) verifica se é final de mapa
        JR NZ,OPEN6
        LD A,#FE      ; ) acinala que não foi achado a entrada
        POP IY
        JP ERRGER
OPEN8  :LD (POSSEQ),IY
        LD (PONTBUF),IY
        POP IY
        LD HL,(DATARQ)            ; ) guarda as variáveis de trabalho em DCB
        LD (IY+12),L
        LD (IY+13),H
        LD HL,(ENTARQ)
        LD (IY+14),L
        LD (IY+15),H
        LD HL,(LOWTAM)
        LD (IY+16),L
        LD (IY+17),H
        LD HL,(HITAM)
        LD (IY+18),L
        LD (IY+19),H
        LD A,(ATRARQ)
        LD (IY+20),A
        LD HL,(POSNOM)
        LD (IY+21),L
        LD (IY+22),H
        LD HL,(POSSEQ)
        LD (IY+23),L
        LD (IY+24),H
        LD HL,(FORMAT)
        LD (IY+25),L
        LD (IY+26),H
        LD HL,(TAMSET)
        LD (IY+27),L
        LD (IY+28),H
        LD HL,(SOMA)
```

```
        LD (IY+29),L
        LD (IY+30),H
        LD HL,(PONTBUF)
        LD (IY+31),L
        LD (IY+32),H
        XOR A
        AND A
        RET

CONVDRV:NOP                     ; << CONVERTE NUMERO DO DRIVE >>
        AND A                   ; > A=#00 - DEFAUT
        JR Z,CONVD01
        CP #FF                  ; > A=#FF - NÃO INTERPRETA
        RET Z
        DEC A                   ; > A>#00 - A=A-1
        LD (DRIVE),A
CONVD01:LD A,(DRIVE)
        CALL TRODSC
        CALL SOCONV
        CALL MNTDIR
        RET

ADMA    :LD (VARCOP),DE         ; << ALTERA DMA (Direct Memory Adjust) >>
        RET

READ    :LD A,(ATRARQ)          ; << LE UM CLUSTER DO ARQUIVO >>
        CP 2
        JP Z,ERRGER
        PUSH DE
        CALL SETVAR
        LD IY,(PONTBUF)         ; > IY com ponteiro para tabela bufpgm
        LD H,(IY+1)
        LD L,(IY+0)
        CALL CONVCLT
        LD HL,(VARCOP)
        LD BC,(FORMAT)
        LD A,(DRIVE)
        AND A
        CALL PHYDIO   ; > carrega um setor
        LD A,#FF
        JP C,ERRGER
        LD HL,(PONTBUF)         ; > próximo cluster
        INC HL
        INC HL
```

```
        LD (PONTBUF),HL
        POP IY
        LD (IY+31),L
        LD (IY+32),H
        LD HL,(VARCOP)           ; > atualiza variável com posição de memória
        LD DE,(TAMSET)
        ADD HL,DE
        LD (VARCOP),HL
        LD IY,(PONTBUF)          ; > pega o próximo cluster e coloca em HL
        LD H,(IY+1)
        LD L,(IY+0)
        LD DE,#FFF   ; > para verifica final de arquivo
        CALL CPHLDE
        LD A,1
        JP Z,ERRGER
        XOR A
        AND A
        RET

SETVAR :PUSH DE                  ; << PEGA AS VARIAVEIS DE TRABALHO DO FCB >>
        POP IY
        LD L,(IY+12)
        LD H,(IY+13)
        LD (DATARQ),HL
        LD L,(IY+14)
        LD H,(IY+15)
        LD (ENTARQ),HL
        LD L,(IY+16)
        LD H,(IY+17)
        LD (LOWTAM),HL
        LD L,(IY+18)
        LD H,(IY+19)
        LD (HITAM),HL
        LD A,(IY+20)
        LD (ATRARQ),A
        LD L,(IY+21)
        LD H,(IY+22)
        LD (POSNOM),HL
        LD L,(IY+23)
        LD H,(IY+24)
        LD (POSSEQ),HL
        LD L,(IY+25)
        LD H,(IY+26)
        LD (FORMAT),HL
```

```
        LD L,(IY+27)
        LD H,(IY+28)
        LD (TAMSET),HL
        LD L,(IY+29)
        LD H,(IY+30)
        LD (SOMA),HL
        LD L,(IY+31)
        LD H,(IY+32)
        LD (PONTBUF),HL
        RET

CONVCLT:LD DE,(SOMA)            ; << CONVERTE CLUSTER EM SETOR P/ PHYDIO >>
        ADD HL,DE
        LD A,(TAMSET+1)
        RRCA
        DEC A
        EX DE,HL
        RET Z
        EX DE,HL
        ADD HL,HL
        EX DE,HL
        RET

CPESC  :LD A,7                  ; << TESTA A TECLA ESC >>
        OUT (#AA),A
        IN A,(#A9)
        CP #FB
        RET NZ
        LD A,#FF

ERRGER :AND A       ; << PROCEDIMENTO GENÉRICO DE ERRO >>
        SCF
        RET

AVCSEQ :CALL SETVAR; << AVANÇA UM CLUSTER NA TRILHA >>
        PUSH DE
        LD IY,(PONTBUF)
        LD L,(IY+2)
        LD H,(IY+3)
        LD DE,#FFF    ; > verifica se chegou ao final da trilha
        CALL CPHLDE
        PUSH IY
        POP HL
        POP DE
```

```
        RET Z
        INC HL                  ; > avança um cluster no ponteiro
        INC HL
        LD (PONTBUF),HL         ; > registra nova posição do ponteiro
        PUSH DE
        POP IY
        LD (IY+31),L  ; > grava ponteiro em FCB
        LD (IY+32),H
        RET

RETSEQ :CALL SETVAR ; << RETORNA UM CLUSTER NA TRILHA >>
        EXX
        LD HL,(PONTBUF)
        LD DE,(POSSEQ)
        CALL CPHLDE   ; > verifica se ponteiro está no inicio
        RET Z
        DEC HL                  ; > retorna um cluster no ponteiro
        DEC HL
        LD (PONTBUF),HL         ; > registra nova posição do ponteiro
        PUSH HL
        EXX
        POP HL
        PUSH DE
        POP IY
        LD (IY+31),L  ; > grava ponteiro em FCB
        LD (IY+32),H
        RET

MSTCLR :CALL SETVAR ; << RETORNA O CLUSTER ATUAL DA TRILHA >>
        LD IY,(PONTBUF)
        LD L,(IY+0)
        LD H,(IY+1)
        RET

MSTTAM :CALL SETVAR ; << RETORNA O TAMANHO DO ARQUIVO ATUAL >>
        LD HL,(HITAM)
        LD DE,(LOWTAM)
        RET

MSTDAT :CALL SETVAR ; << RETORNA A DATA DO ARQUIVO ATUAL >>
        LD HL,(DATARQ)
        RET

DATARQ :DEFW 0
ENTARQ :DEFW 0
LOWTAM :DEFW 0
HITAM  :DEFW 0
ATRARQ :DEFB 0
POSNOM :DEFW 0
POSSEQ :DEFW 0

PONTBUF:DEFW 0
[illegible] :DEFW 0
```

```
Para solucionar o erro no fonte, modifique a rotina SQC009 para:

SQC009 :LD A,(VSET)      ; << VERIFICA E TRATA CLUSTER APAGADO >>
        CALL STCONV
        LD A,H
        OR L
        PUSH AF
        EX AF,AF'
        POP AF
        POP IX
        CALL Z,SQC008    ; > INCLEMENTA PONTEIROS
        PUSH IX
        EX AF,AF'
        JR Z,SQC009
        XOR A
        RET

E troque o início da rotina ERRMAP de:

ERRMAP :LD HL,(SOMA)     ; << REGISTRA INFORMAÇÕES P/ DISCO VASIO >>
        INC HL
        INC HL
        LD DE,4
        AND A
        SBC HL,DE        ; > PEGA 1º CLUSTER LIVRE

Para:

ERRMAP :LD HL,2          ; << REGISTRA INFORMAÇÕES P/ DISCO VASIO >>


Para solucionar o erro na listagem 1, alterar as seguintes linhas para

9000 C3 2C 90 AF C3 74 90 C3 13 91 C3 8B 92 C3 70 92

9160 26 90 CA 70 92

9250 FF FF CD 95 91 DD E1 C9 3A 18 94 CD 2C 90 7C B5
9260 F5 08 F1 DD E1 CC 1D 92 DD E5 08 28 EB AF C9 00
9270 21 02 00
```

LISTAGEM 6 — Errata da parte II deste arquivo

```
1 '
2 '
3 ' Programa: EXAFAT-EXAminador da FAT
4 ' Direitos: Revista CPU
5 '      Por: J+lio Velloso
6 '   Versxo: 40 colunas
7 '
1000 SCREEN0:WIDTH40:COLOR,1,1:KEYOFF:CLEAR1200,&HA000:DEFINTA-Z:LOCATE,,0:GOSUB 1520
1010 '
1020 BLOAD"DRIVE2.BIN"
1030 '
1040 DEFUSR1=&H9000:DEFUSR2=&H9003:DEFUSR3=&H9007:DEFUSR4=&H900A:DEFUSR6=&H900D:DEFUSR7=&H9010:DEFUSR8=&H9013:DEFUSR9=&H9016:DEFUSR0=&H9019
1050 '
1060 A=USR2(0):A=USR3(0):A=USR4(0):A=USR7(0):GOSUB1630
1070 '
1080 BPGN!=256*PEEK(&H901D)+PEEK(&H901C):BDIR!=256*PEEK(&H901F)+PEEK(&H901E):DIR!=256*PEEK(&H9021)+PEEK(&H9020):FAT!=256*PEEK(&H9023)+PEEK(&H9022):BFAT!=256*PEEK(&H9025)+PEEK(&H9024)
1090 '
1100 GOSUB 1790
1110 '
1120 PGN!=BPGN!+2
1130 '
1140 ADIR!=DIR!:SE!=256*PEEK(PGN!+1)+PEEK(PGN!):IF SE!=65535! GOTO 1470
1150 CI!=PEEK(ADIR!+27)*256+PEEK(ADIR!+26)
1160 IF CI!<>SE! THEN ADIR!=ADIR!+32:IF PEEK(ADIR!)<>0 GOTO 1150
1170 '
1180 LOCATE10,6:FORI=0TO7:PRINTCHR$(PEEK(ADIR!+I));:NEXTI:PRINT".";:FORI=8TO10:PRINTCHR$(PEEK(ADIR!+I));:NEXTI
1190 '
1200 LOCATE31,6:PRINTPEEK(ADIR!+29)*256+PEEK(ADIR!+28)+PEEK(ADIR!+31)*4096+PEEK(ADIR!+30)*65536!;"   ";:LOCATE38,6:PRINT T2$" "T2$
1210 '
1220 LOCATE7,8:DAT!=PEEK(ADIR!+25)*256+PEEK(ADIR!+24):PRINT(DAT! AND &H1F);"/";(DAT! AND &H1E0)/32;"/";((DAT! AND &HFE00)/256)+74
1230 '
1240 NN=1:NY=12:LOCATE 2,NY
1250 SE=256*PEEK(PGN!+1)+PEEK(PGN!):IF SE=&HFFF GOTO 1300
1260 PRINT (SE+PEEK(&H940C))*2;" ";:PGN!=PGN!+2
1270 NN=NN+1:IF NN=7 THEN NN=1:NY=NY+1:LOCATE2,NY
1280 GOTO 1250
1290 '
1300 A$=INKEY$:IF A$="" GOTO 1300
1310 IF ASC(A$)=28 THEN PGN!=PGN!+2:LOCATE3,12:A=USR(0):GOTO 1140
1320 IF ASC(A$)=29 GOTO 1400
1330 IF ASC(A$)=30 THEN LOCATE3,12:A=USR(0):GOTO 1120
1340 IF ASC(A$)=31 GOTO 1450
1350 IF ASC(A$)=27 THEN CLS:KEY ON:END
1360 IF ASC(A$)=12 THEN LOCATE3,12:A=USR(0):GOTO 1030
1370 IF ASC(A$)=24 GOTO 1680
1380 GOTO 1300
1390 '
1400 GOSUB 1490
1410 IF PGN!=BPGN! GOTO 1430
1420 GOSUB 1490
1430 PGN!=PGN!+2:LOCATE3,12:A=USR(0):GOTO 1140
1440 '
1450 LOCATE3,12:A=USR(0)
1460 PGN!=PGN!+2:SE!=PEEK(PGN!+1)*256+PEEK(PGN!):IF SE!<>65535! GOTO 1460
1470 PGN!=PGN!-2:GOSUB 1490:PGN!=PGN!+2:GOTO 1140
1480 '
1490 PGN!=PGN!-2:SE!=PEEK(PGN!+1)*256+PEEK(PGN!):IF SE!<>4095 GOTO 1490
1500 RETURN
1510 '
1520 DEFUSR=&H41:A=USR(0):T1$=CHR$(1)+CHR$(64+23):T2$=CHR$(1)+CHR$(64+22):T3$=CHR$(1)+CHR$(64+24):T4$=CHR$(1)+CHR$(64+25):T5$=CHR$(1)+CHR$(64+26):T6$=CHR$(1)+CHR$(64+27):T7$=CHR$(1)+CHR$(64+20):T8$=CHR$(1)+CHR$(64+19)
1530 LOCATE2,1:PRINT"Programa: EXAFAT"
1540 LOCATE5,3:PRINT"Autor: J+lio Velloso"
1550 LOCATE29,1:PRINTCHR$(&HC7)CHR$(&HD3)" "CHR$(&HC7)CHR$(&HC1)" "CHR$(&HD0)CHR$(&HDE)
1560 LOCATE29,2:PRINTCHR$(&HD0)"  "CHR$(&HC1)CHR$(&HC7)CHR$(&H20)CHR$(&HD0)CHR$(&HDE)
1570 LOCATE29,3:PRINTCHR$(&HC1)CHR$(&HD6)" "CHR$(&HD0)"  "CHR$(&HC1)CHR$(&HC7)
1580 FORI=0TO38:LOCATEI,4:PRINTT1$;:LOCATEI,0:PRINTT1$;:LOCATEI,24:PRINTT1$;:IFI<25THENLOCATE0,I:PRINTT2$;:LOCATE38,I:PRINTT2$;:NEXTIELSENEXTI
1590 GOSUB 1820:FORI=0TO38:LOCATEI,19:PRINTT1$:NEXT:LOCATE0,19:PRINTT7$:LOCATE38,19:PRINTT8$
1600 LOCATE0,0:PRINTT3$;:LOCATE38,0:PRINTT4$;:LOCATE0,24:PRINTT5$;:LOCATE38,24:PRINTT6$;:LOCATE0,4:PRINTT7$;:LOCATE38,4:PRINTT8$;:DEFUSR=&H44:A=USR(0):DEFUSR=&HBA00
1610 RETURN
1620 '
1630 DEFUSR=&HBA00:RESTORE1650:FORI=&HBA00TO&HBA1F:READA$:POKEI,VAL("&h"+A$):NEXTI
1640 RETURN
1650 DATA 06,07,C5,3A,DD,F3,F5,06,23,C5,3E,20,CD,A2,00,C1,10,F7,21,DC,F3,7E,3C
1660 DATA 77,F1,32,DD,F3,C1,10,E3,C9
1670 '
1680 CLS:LOCATE0,0:FILES:FORI=0TO38:LOCATEI,21:PRINTT1$:NEXTI:LOCATE2,22:LINEINPUT"Entre com o arquivo: ";A$
1690 CADIR!=ADIR!:ADIR!=DIR!
1700 J=LEN(A$)
1710 BAS!=ADIR!:FORI=1TOJ:IFMID$(A$,1,I)="."THENBAS!=ADIR!+8:I=I+1
1720 IFPEEK(BAS!)=ASC(MID$(A$,I,1))THENBAS!=BAS!+1:NEXTI:GOTO 1750
1730 ADIR!=ADIR!+32:IFPEEK(ADIR!)<>0GOTO 1710
1740 LOCATE2,22:PRINT"Arquivo nxo encontrado";SPC(15):FORI=1TO1500:NEXTI:ADIR!=CADIR!
1750 CI!=PEEK(ADIR!+27)*256+PEEK(ADIR!+26):PGN!=BPGN!
1760 SE!=PEEK(PGN!+1)*256+PEEK(PGN!):IFSE!<>CI!THENPGN!=PGN!+2:GOTO1760
1770 CLS:GOSUB1520:GOSUB1790:GOTO 1170
1780 '
1790 LOCATE 2,6:PRINT"Arquivo:":LOCATE 23,6:PRINT"Tamanho: ":LOCATE 2,8:PRINT"Data:":LOCATE 2,10:PRINT"Distribuixxo:"
1800 RETURN
1810 '
1820 LOCATE1,20:PRINT"SLCT/CLS/ESC - examina/inicializa/sai"
1830 LOCATE1,21:PRINT">/<  - avanxa/retorna arquivo"
1840 LOCATE1,22:PRINT"^/v  - primeiro/ultimo arquivo"
1850 RETURN
```

LISTAGEM 7

# F-16 FIGHTING FALCON

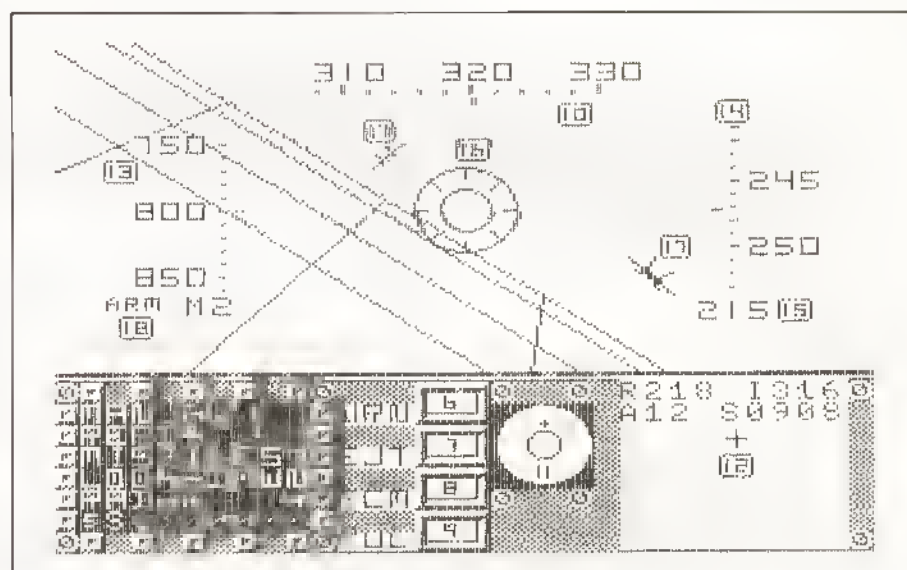
ANDRÉ LUIZ ANCIÃES DOS SANTOS

### INTRODUÇÃO

Este jogo é o simulador de vôo da aeronave F-16 Fighting Falcon, uma máquina com excelente desempenho, para pilotos de alto nível de qualificação. Essa nave, além de possuir misseis e uma metralhadora de 20mm, conta com sistemas automáticos de operação.

Seu objetivo é combater os Mig-25 soviéticos, que possuem armamentos e equipamentos similares aos do F-16. Sua nave pode alcançar 1970 milhas por hora (mp) e altitude máxima de 50.000 pés.

### COMANDOS DA NAVE

Subir — Cursor (Joystick) p/baixo
Descer — Cursor (Joystick) p/cima
Esquerda — Cursor (Joystick) p/esquerda
Direita — Cursor (Joystick) p/direita
X — Aumentar a velocidade
Z — Diminuir a velocidade

SHIFT — Disparo de missil e metralhadora
A — Acelera bruscamente
Z e X — Diminui bruscamente (pressionar as 2 teclas juntas)

**COMANDOS DO EQUIPAMENTO**

C — muda de semi-automático para manual e vice-versa
F3— muda de manual para automático e vice-versa
F2 — muda o feixe de radar de Look Up para Look Down e vice-versa.
F1 — Troca de mísseis para metralhadora e vice-versa
HOME — Termina o jogo com o piloto salvo
STOP — Pausa do jogo
ESC — Paralisa o jogo sem score.
OBS: Estando em modo automático, a tecla C será ignorada.

**TELA DO JOGO**

1 — Piloto automático, semi-automático e manual
2 — Indicador de condição de uso do armamento (Ready)
3 — Indicador do armamento utilizado
4 — Estado do radar: Look Up ou Look Down
5 — Score
6 — Luz de alerta contra disparo de missil inimigo
7 — Ejetor do piloto (esse comando é automático)
8 — Sistema contra medidas eletrônicas (ECM System)
9 — Indicador de combustivel
10— Bússola (0 a 360 graus)
11— Radar de curto alcance
12— Radar de longo alcance
13— Velocidade em milhas por hora (MPH)
14— Altura medida em pés
15— Quando disparado o missil, marca o tempo de impacto no alvo. Antes do disparo, marca a distância do alvo.
16— Mira da aeronave (a mira do missil é menor)
17— Naves inimigas
18— Quantidade de armamento disponivel.

**DICAS**

(1) O missil só poderá ser disparado a uma distância minima de 300m. A mira então ficará vermelha indicando que o alvo poderá ser atingido.
(2) Se o alvo estiver muito perto, troque o armamento para metralhadora. Será mais fácil de atingir o inimigo.
(3) Não leve muito tempo para destruir os inimigos, pois eles se posicionarão atrás de uma nave para tentar destrui-lo.
(4) A tecla A é de alta eficiência. Quando pressionada, aumenta a velocidade bruscamente, permitindo a saida rápida do alcance do fogo inimigo.
(5) Somente utilize as mudanças bruscas de velocidade em caso de fuga do inimigo, pois o consumo de combustivel aumentará, diminuindo seu tempo de vôo.
(6) No caso da metralhadora, somente dispare quando o meio da mira ficar vermelho. São necessários muitos tiros para abater o inimigo.
(7) Não mantenha a nave estabilizada por muito tempo, para não ser facilmente localizado pelo inimigo.
(8) Quando o inimigo estiver disparando atrás da sua nave, você ouvirá um alarme. Imediatamente mude o curso da nave para não ser atingido.
(9) Quando disparar um missil, mantenha a nave inimiga na mira, pois só assim o seu missil atingirá o alvo.

**OUTROS DETALHES**

— A troca de armamentos leva 32 segundos para se processar, e enquanto isso nenhuma outra arma poderá ser utilizada.
— O radar em modo Look Down, entre 3.000 e 13.000 pés de altitude, tem alcance de 60 milhas.
— O radar em modo Look Up, entre 13.000 e 30.000 pés de altitude, tem alcance de 80 milhas.
R — Significa distância do alvo;
I — Rumo do alvo.
A — Significa altitude de alvo;
S — Velocidade do alvo.

— O F16 FF está preparado para voar ao nivel do mar. Dentro do radar de curto alcance, aparecerão todas as naves inimigas num raio de 32 milhas.

# LICENCE TO KILL

ANDRÉ ANCIÃES DOS SANTOS

Incorporando James Bond, o agente 007, você deve desbaratar uma quadrilha de traficantes de tóxicos, cumprindo 5 fases:

### FASE I

Na primeira fase você, pilotando um helicóptero, deve chegar até a base de operações da quadrilha. Enquanto voa até lá, pode tentar destruir um carro que anda sob o seu helicóptero. Porém, caso você não o destrua, apenas deixará de ganhar alguns pontos.

Nas laterais das ruas, existem quadrados pretos, de onde saem tiros. Você pode destruí-los, bastando para isso atirar neles.

Nesta fase, os controles são os seguintes:

UP - Acelera o helicóptero e diminui e altitude

DOWN - Diminui a velocidade e aumenta a altitude

RIGHT - Move para a direita

LIFT - Move para a esquerda

FIRE - Atira

Se você tocar as casas com o helicóptero, perderá uma vida. Todo o resto dos objetos é inofensivo, desde que você mantenha a altitude acima do segundo traço, de baixo para cima no mostrador (veja "TELAS DE JOGOS").

O "DAMAGE", como em todas as outras fases, mostra o quanto seu helicóptero foi danificado até agora. Se a barra vermelha, abaixo da palavra "DAMAGE", chegar até o lado direito, você perderá uma vida.

### FASE 2

Chegando no Q.G., você tem agora que atravessá-lo, procurando não ser atingido.

Para facilitar a sua tarefa, foi desenhado um mapa dessa fase, publicado junto das Instruções.

Para terminar a fase, você deve ultrapassar a linha pontilhada no topo do mapa. Os controles são usados para mover Bond, e o FIRE atira na direção para a qual sua mira estiver apontada. Para mover a

mira, faça o seguinte: Aperte FIRE, e, mantendo-o pressionado, mova para a esquerda ou para a direita. Você verá um círculo se mover em torno de James. Ele é a mira. Quando ela estiver na posição desejada, solte o FIRE, e o tiro será disparado.

No mapa, as partes hachuradas são objetos e casas, por onde é impossível passar. Os círculos pretos são barris. Para destruí-los, é necessário acertar quatro tiros. Ao serem destruídos, os barris explodem, matando os inimigos que estiverem próximos a ele.

Um ponto importante é procurar sempre economizar balas. Uma dica: quando você estiver com três cartuchos (vide "TELA DE JOGO") e aparecer algum outro na tela (ao matar os inimigos, às vezes aparecem cartuchos), antes de apanhá-lo, gaste todos os tiros do terceiro cartucho. Dessa forma, você aproveita os quinze tiros do novo.

### FASE 3

Nesta fase, a mais fácil do jogo, você deve, estando dependurado num helicóptero por uma corda, entrar em um hidroavião.

Não há muito o que dizer, apenas leve Bond até o avião e aperte FIRE.

### FASE 4

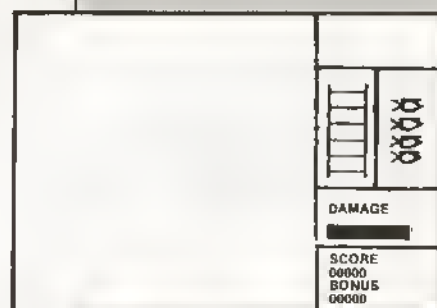
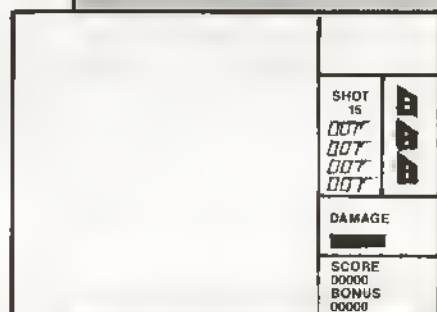
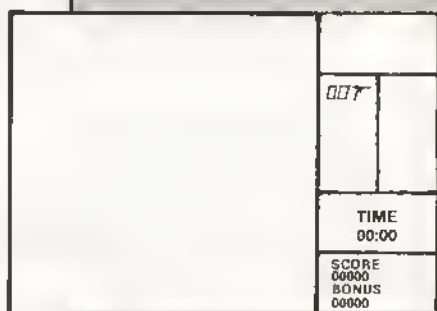
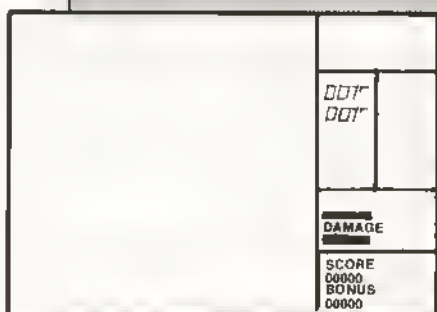
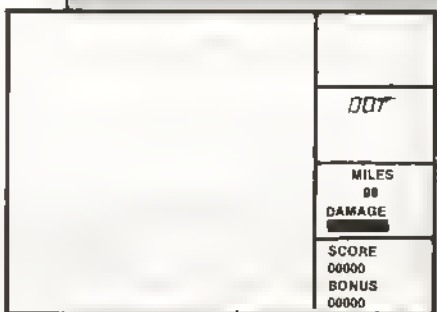
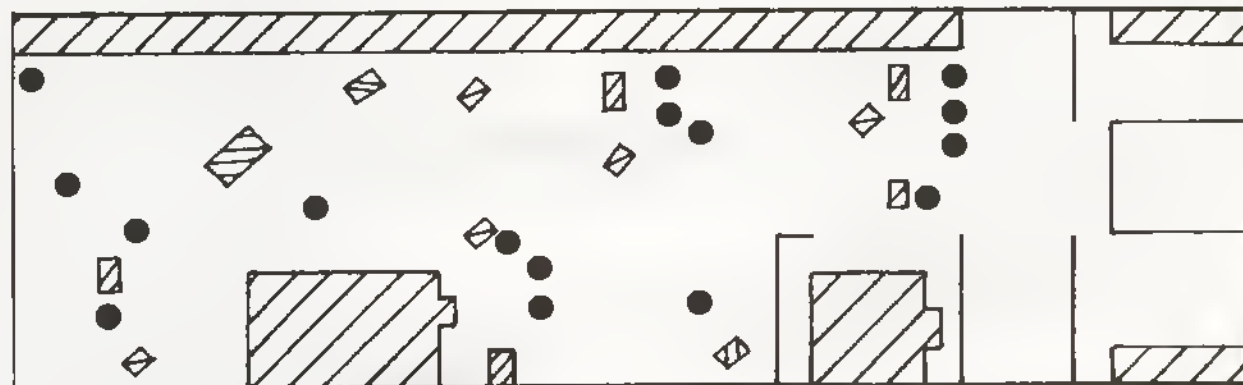
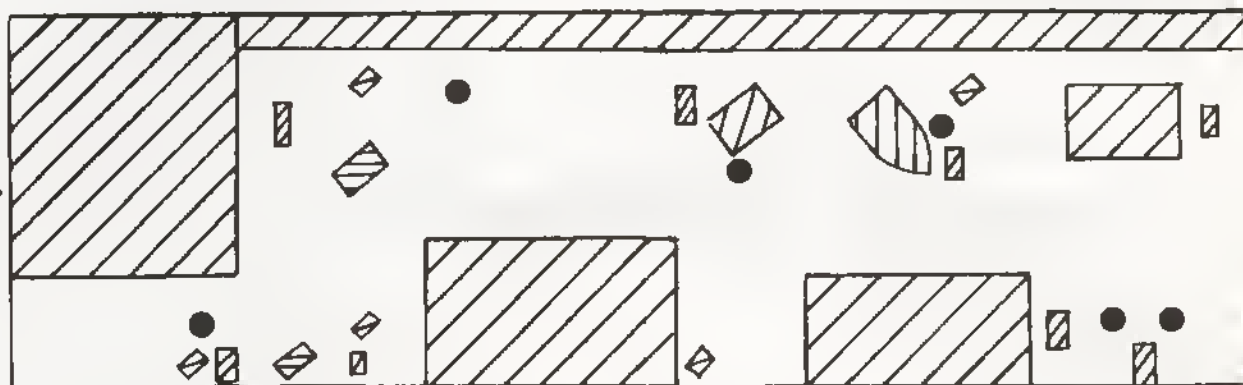
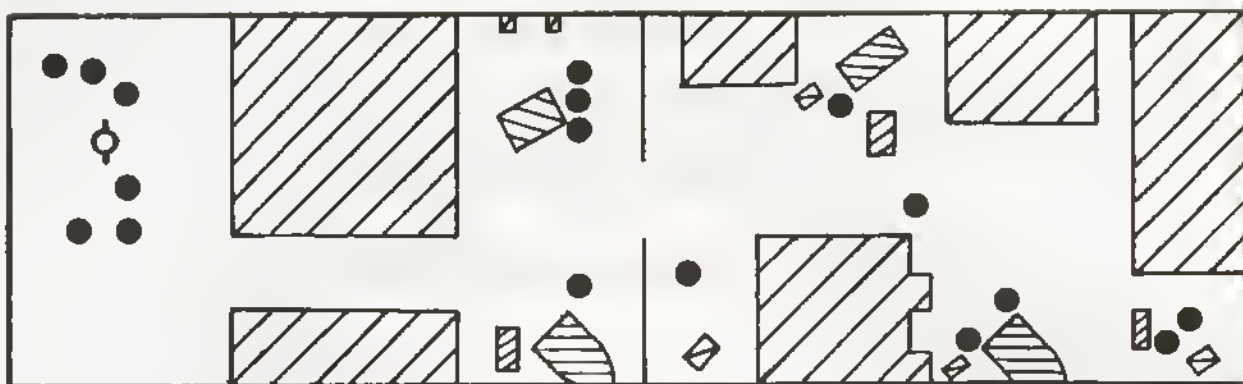
Agora, você está na água, e deve acertar um avião com um arpão, subindo por uma corda até ele.

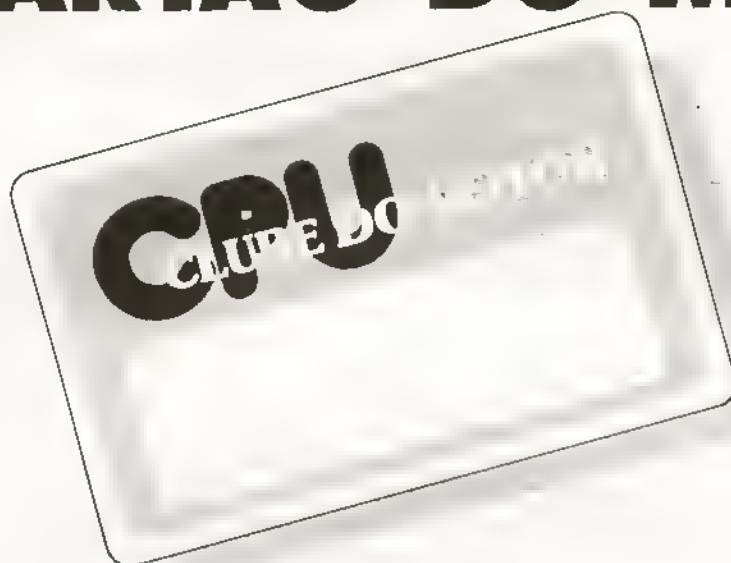
Para conseguir o arpão, deve matar um dos mergulhadores. Para matá-los, você deve estar sob a água, passar sobre eles e apertar FIRE. Colocando o joystick para a frente, Bond ficará submerso. Cuidado para o oxigênio não acabar (VIDE "TELA DE JOGO").

Existem alguns pacotes de drogas na água. Se você destruí-los, ganhará alguns pontos extras.

Para pegar o avião, você deve acertar em um dos flutuadores existentes (em forma de Zeppelin, presos às asas), e, quando o arpão estiver passando sobre eles, teclar FIRE. Aparecerá uma corda entre você e o avião. Você então deve chegar até ele , desviando das pedras existentes na água. Ao chegar ao avião, automaticamente passará para a última fase.

### FASE 5

Nesta fase, você deve jogar cinco caminhões para fora da estrada. Colocando o joystick para a frente, seu caminhão acelerará. Então, recue aos poucos para não perder velocidade, e espere o caminhão inimigo aparecer. Quando ele chegar bem perto, corte-o pelo lado, ultrapasse-o e então empurre-o para fora, utilizando para isso sua carroceria. Não bata com a parte da frente, ou isso aumentará o "damage" Após jogar cinco caminhões para fora da estrada, será mostrada uma mensagem de congratulações, e o jogo se reiniciará.

INÍCIO
LICENCE TO KILL

# CLUBE DO LEITOR
# O CARTÃO DO MSX

**CANAL TRÊS INFORMÁTICA**
30% desconto na inscrição do clube do MSX (MSXCLUBE)
15% desconto software (jogos em geral)
Facilidade no pagamento na compra de periféricos

**MANÍACOS OO MSX**
15% desconto na compra de jogos.
20% desconto na compra de jogos especiais.
10% desconto na compra de programas de autores nacionais
15% desconto na compra de aplicativos.

**CONECTOR IND. E COM. LTDA**
5% desconto na compra de kit de drive para MSX.

**CIBERTRON ELETRÔNICA LTOA**
15% desconto na compra de software

**YOUNGSOFT**
**30% desconto nas compras de sottwere.**
**10% desconto ne Inscrição no clube de usuárlos.**

**NEMESIS INFORMÁTICA LTDA**
10% desconto em seus produtos.

**RECURSOS DIGITAIS**
5% desconto na compra de perilérícos.
10% desconto na compra de solts de outras empresas.
30% desconto na compra de softs da Redi Universoft.

**TACTO INORMÁTICA COM. LTOA**
10% desconto na compra de qualquer produto ou curso.

**PAULISOFT INFORMÁTICA LTOA**
10% desconto na compra de software, exceto promoção.

**OISCOVERY INFORMÁTICA**
10% desconto em seus produtos.

**EDITORA ALEPH**
15% desconto em suas publicações

**REVOLUTION**
20% desconto na compra de software

**NEWSOFT**
10% desconto na compra dos jogos comuns.
20% desconto nos jogos especiais.
25% desconto nos aplicativos.
30% desconto na compra de livros
5% desconto na compra de perifèricos e suprimentos.
• Os descontos acima não incidirão sobre produtos em promoção.

**NEWOATA**
***5% desconto nos produtos de representação/revende.**
**10% desconto nos seus produtos.**

**ESPACIAL ELETRÔNICA**
20% desconto nos seus produtos.

**INFORTELLES**
5% desconto em gerat

**GAME OF TIME**
10% desconto em geral

**SOFTMARK**
12% desconto nos seus produtos.

SOFT DESIGNS
15% desconto na compra de software e serviços

**MSX INFORMÁTICA**
10% desconto em hardware
20% desconto em software da MSX INFORMÁTICA e ou
10% desconto em software de outras EMPRESAS.
10% desconto em assistência técnica e suprimentos.

**A&A SOFTWARE**
25% desconto na compra de jogos
15% desconto em software da A&A SOFTWARE
10% desconto em software de outras empresas
5% desconto em suprimentos

**BITCENTER INFORMÁTICA**
10% desconto em qualquer serviço de manutenção

Se você ainda não tem um cartão,
faça logo a sua assinatura de CPU e receba o seu!

# JOE BLADE

O que você acha de ser um prisioneiro nazista, com a missão de resgatar os outros prisioneiros, acionar uma bomba e fugir antes que ela exploda? Na vida real esta situação não seria nada interessante, mas em seu MSX é uma emocionante aventura.

André Luis Anciães dos Santos

## O JOGO

Como já foi dito, o seu objetivo é salvar os prisioneiros, ativar a bomba e fugir.

Os prisioneiros se encontram espalhados pelas salas do jogo. Eles não tem lugar fixo, aparecendo em salas aleatórias, sorteadas no início de cada partida.

Existem vários objetos para ajudá-lo a conseguir seu intento:

– MUNIÇÃO (AMMUNITION) – Têm o aspecto de uma caixa. Servem para recarregar a sua arma. Seus tiros, embora não tenham um marcador na tela, são limitados.
– COMIDA (FOOD) – Parece um lanche gigante, com direito a coca-cola. Recoloca sua energia no máximo.
– UNIFORME INIMIGO (ENEMY UNIFORM) – Parece um quepe. Deixa você imune aos inimigos.

Além desses objetos, ainda existem as chaves. Existem algumas portas com grades que só podem ser abertas usando uma chave. Cada vez que você passar por uma destas portas, usará uma chave.

O jogo se inicia na tela onde existe um 'I' maiúsculo. As portas são os buracos nas linhas. Os buracos com rachuras representam as portas que devem ser abertas com uma chave. Vale ressaltar que só é necessário usar a chave para "subir" no mapa.

Nas portas onde existem círculos, você, ao descer, deve fazer o seguinte: vire para a direita, desça e atire imediatamente.

Abaixo, um pequeno roteiro para terminar o jogo:
– Primeiro, procure os prisioneiros, em todas as salas.
– Após percorrer todas as salas, procure uma bomba. Ao encostar nela, aparecerão as letras de A a E fora de ordem. Você deve colocá-las na ordem em 30 segundos.
– Vá à tela inicial e saia.

## DICAS

– Após ativar a bomba, você tem 20 minutos para sair do quartel.
– Cuidado para não ficar preso em alguma sala sem chaves.
– Procure economizar munição, pulando os inimigos.
– Não pegue a bomba no início do jogo, para não se arriscar com o tempo.

# JOE BLADE

SÉRGIO PAIXÃO MORGADO

## OH! SHIT! VIDA INFINITA

```
20 BLOAD"OHSHIT1.BIN"
30 FORA=&HB34BTO&HB353·POKEA,0:NEXT
40 DEFUSR=&HD000:A=USR(0)
50 BLOAD"OHSHIT2,BIN"
60 DEFUSR=&HD000·A=USR(0)
```

## FREDDY HARDEST IN SOUTH MANHATTAN ENERGIA INFINITA

```
20 POKE-1,(NOT(PEEK(-1))AND&HF0)*1.0625
30 IFPEEK(&HF342)=&H8F THEN POKE-609,201
40 COLOR 1,1,1:SCREEN2
50 CLEAR 100,35000!
60 BLOAD"FREDDYH1",R:BLOAD"FREDDYM2",R
70 BLOAD"FREDDYM3",R:BLOAD"FREDDYM4",R
80 BLOAD"FREDDYM5",R:BLOAD"FREDDYM6",R
90 BLOAD"FREDDYM7",R:BLOAD"FREDDYM8":
POKE&HA00D,0:POKE&HA00E,0:POKE&HA00F,0
100 DEFUSR=PEEK(-832)*256+PEEK(-833)
110 FORF=0TO1500:NEXT:X=USR(0)
```

## GONZZALES 1 VIDA INFINITA

```
10 REM POKE BY SERGIO P  MORGADO
20 A=INP(&HA8)/16ORINP(&HA8)POKE-
2,A:POKE1,(NOT(PEEK(-1))AND&HF0)*1.0625
30 IF PEEK(&HF342)=&H8FTHENPOKE-609,201
40 BLOAD"GONZA11",R
50 BLOAD"GONZA12".POKE&HB764,0:
DEFUSR=&HC800:A=URS(0)
60 BLOAD"GONZA13",R
70 BLOAD"GONZA14",R
```

## ZANAC 3 VIDA INFINITA E VELOCIDADE DO JOGO

```
20 CLS:KEYOFF·POKE&HD000,0
21 LOCATE7,0·PRINT"ALTERAR VELOCIDADE DO
JOGO"
22 LOCATE10,4·PRINT"0 - VELOCIDADE
RAPIDA"
23 LOCATE10,6 PRINT"1 - VELOCIDADE
NORMAL"
24 LOCATE10,8·PRINT"2 - VELOCIDADE
LENTA"
25 LOCATE10,10·PRINT"3 - VELOCIDADE MAIS
LENTA"
26 LOCATE10,16·PRINT"OPÇÃO";:INPUTB
27 IFB>3THENLOCATE17,16 PRINT" "·GOTO26
30 DEFUSR=&HC000 A$="ZANAC 3"
40 BLOAD"ZANAC31"·X$=USR(A$) SCREEN2
50 BLOAD"ZANAC32" POKE&H8B0D,B
POKE&H8E54,0:DEFUSR=34775:
60 POKE 35417!,PEEK(&HD000)
71 X=USR(I)
80 BLOAD"ZANAC33" R
```

## MUTANT ZONE 1 VIDA INFINITA E IMUNIDADE

```
30 A=INP(&HA8)/16 OR INP (&HA8) POKE-
2,A:POKE -1,(NOT(PEEK(-1)) AND
&HF0)*1 0625
40 BLOAD "MUTANT11"
50 BLOAD"MUTAN12":DEFUSR=&HFA00.X=USR(0)
60 BLOAD"MUTAN13":FOR A=&HC126 TO &HC13D
: POKE A,0 : NEXT : DEFUSR = &HFA29 :
X=USR(0)
70 BLOAD"MUTANT14"·DEFUSR=&HFA47·
X=USR(0)
80 BLOAD"MUTANT15"·POKE &H9C19,0   FOR
F=0 TO 1500 : NEXT :
DEFUSR=&HFA5E:X=USR(0)
90 'OBS  PARA MUDAR A COR DO JOGO
PRESSIONAR UM DOS NUMEROS DE 1 A 9
```

## JAWS (TUBARÃO) VIDA INFINITA

```
20 POKE-2,INP(&HA8)/16ORINP(&HA8) POKE·
1,(NOT(PEEK(-1)AND&HF0)*1.0625
30 IFPEEK(&HF342)=&H8F THEN POKE -
609,201
40 KEYOFF·COLOR1,1,1.SCREEN2
50 BLOAD"JAWS2":DEFUSR=&HA388:X:USR(0)
60 BLOAD"JAWS3"·BLOAD"JAWS4"
DEFUSR=&HFA00:X=USR(0)
70 BLOAD"JAWS":DEFUSR=&HFA17:X=USR(0)
80 BLOAD"JAWS6"·POKE&H946D,0:
POKE&H946E,0:POKE&H946F,0·DEFUSR=&HFA2E:
X=USR(0)
90 BLOAD"JAWS7":FORF=0TO2000 NEXT:
DEFUSR=&HFA45:X=USR(0)
```